

AI Workflow Design

From Emergence to Renewal

John Rector

First Edition Draft · 2026

AI Workflow Design
From Emergence to Renewal

First Edition Draft

Copyright © 2026 John Rector
All rights reserved.

This book is published as a free download for educational
and professional use. No portion may be reproduced for commercial
purposes without written permission from the author.

jsrector@gmail.com

Contents

Preface	9
What This Book Is Not	9
What This Book Is	10
Who This Book Is For	10
How to Use This Book	10
A Note on the Recurring Examples	10
The Moment Before the Workflow	11
The Central Thesis	12
The Five-Method Life Cycle	12
The Equation	13
How This Book Is Organized	13
Before You Can Design Anything, You Must Know What You Are Designing For	14
Opening Thesis	15
The Sound Wave	15
What “Unfinished” Means	15
Why Old Workflow Design Begins Too Late	16
The Golf Analogy: Do Not Help the Ball	17
What the Designer’s Job Actually Is	18
The Temptation to Skip Emergence	18
Common Designer Mistake	19
Design Practice	19
Key Takeaways	20
Opening Thesis	20
What an Interface Is	21
What Translation Means	21
The Discipline Shift	22
The Risk of Interface Thinking in AI Contexts	22
AI Does Not Need You to Make It Conversational	23
What “Leaning Into the Nature of AI” Means in Practice	24
Common Designer Mistake	24
Design Practice	25
Key Takeaways	25
Opening Thesis	26
The Equation	26
Why “Messy” Is Not “Wrong”	27
The Clean Solution Statement	28

The Five Recurring Examples	28
What Method 1 Is, and Is Not	30
Common Designer Mistake	31
Design Practice	32
Key Takeaways	33
Opening Thesis	33
Pattern	34
Authority	34
The Four Examples: Pattern and Authority	35
False Completion: A Precise Definition	36
The Four False Completion Examples	37
Status Language as the Production-Safe Solution	38
Authority Maps	39
The Distinction Between Discovery and Production	40
Common Designer Mistake	40
Design Practice	41
Key Takeaways	42
Chapter 5: Overview of the Life Cycle	44
Opening Thesis	44
The Five Methods	44
Why the Order Matters	46
The Recursive Nature of the Life Cycle	47
Two Modes of Life Cycle Use	47
Failure Modes by Skipped Method	48
Common Designer Mistake	49
Design Practice	49
Key Takeaways	50
Chapter 6: Method 1 — Emergence Gives You the Solution	51
Opening Thesis	51
What Emergence Means	51
What Method 1 Is Not	52
The AI's Natural Advantage in Emergence	53
The Golf Analogy: Do Not Help the Ball	53
The Five Examples and Their Solution Statements	54
How to Identify the Solution Statement	55
Full Example: A Live Emergence Transcript	55
Common Designer Mistake	57
Design Practice	58
Key Takeaways	59
Chapter 7: Method 2 — Hallucination Gives You the Ideal	61
Opening Thesis	61
What Hallucination Means in This Context	61
The Critical Constraint: Keep the AI Blind	62
The Production/Discovery Distinction	63
The Demo Mode Prompt	63

What the Output Looks Like	64
Full Example: Blind Ideal Simulation for the Possible Double Charge	65
Common Designer Mistake	66
Design Practice	67
Key Takeaways	68
Chapter 8: Method 3 — As-Is Discovery Gives You Reality	70
Opening Thesis	70
The To-Be Model as Interview Guide	70
What to Gather	71
Discovery vs. Judgment	72
Discovery Interview Structure	72
Full Example: As-Is Discovery for a Restaurant’s Lost-Item Process	73
Common Designer Mistake	75
Design Practice	76
Key Takeaways	77
Chapter 9: Method 4 — To-Be Selling Gives You Transformation	78
Opening Thesis	78
The Commodity Provider vs. The Serious Designer	78
What the Designer Is Not Selling	79
What the Designer Is Selling	79
Business Process Changes the To-Be May Require	80
How to Present the Gap Without Insulting the Client	81
The Transformation Target	81
The To-Be Selling Conversation Structure	82
Full Example: To-Be Selling Conversation for a Restaurant’s Lost-Item and Complaint Work- flow	83
Common Designer Mistake	84
Design Practice	85
Key Takeaways	86
Chapter 10: Method 5 — Stewardship Gives You Renewal	87
Opening Thesis	87
The Maintenance/Stewardship Distinction	87
Two Kinds of Stewardship	88
The Steward’s Structural Position	90
Full Example: Lost-Phone Workflow Stewardship, One Year After Deployment	91
Common Designer Mistake	92
Design Practice	93
Key Takeaways	94
Part Three: Field Methods	95
From Framework to Practice	95
Opening Thesis	96
What an Emergence Session Is	96
The Observer Role	97
Signs of Premature Collapse	98

Transcript Annotation for Emergence Analysis	98
Identifying the Solution Statement from a Transcript	99
Avoiding Over-Intervention and Under-Constraint	99
Running Emergence Sessions with Clients	100
Annotated Transcript Example: Possible Double Charge	100
Common Designer Mistake: Confusing Emergence Data with Requirements	102
Design Practice	103
Key Takeaways	103
Opening Thesis	104
The Simulation Setup	104
The Blind Constraint	105
Why This Produces Better Output	105
The Full Demo Prompt Template	106
Reading the Simulation Output	106
To-Be Model Structure	107
Variance Runs	107
Extracting Authority Checkpoints from Simulation Output	108
Full Example: Bad Restaurant Experience Simulation	108
Common Designer Mistake: Over-Specifying the Problem Statement	110
Design Practice: Sales Call to Decision Maker	111
Key Takeaways	112
Opening Thesis	112
The To-Be Model as Interview Guide	113
The Five Categories of As-Is Discovery	113
Interview Structure and Technique	115
As-Is Map Components	115
Gap Annotation	116
Document Structure	116
Full Example: Restaurant Lost-Item and Complaint Process	117
Common Designer Mistake: Accepting the Official Process as the Full Truth	119
Design Practice: As-Is Discovery for the Possible Double Charge Case	119
Key Takeaways	121
Opening Thesis	121
What the Designer Is Selling	122
The Gap Presentation	122
The Key Question in the Room	123
Why Most of the Value Is in Process Change, Not AI Capability	123
Process Changes to Propose	124
How to Handle Client Resistance	125
The Transformation Target	125
Full Example: Selling the To-Be Model to a Restaurant Chain	126
Common Designer Mistake: Letting the Client Drive the Target to Near As-Is	127
Design Practice: Selling the To-Be Model to a Founder	128
Key Takeaways	129
Opening Thesis	130
The Authority Map	130
The Status Language Library	131
Forbidden Claims	132

Trust Architecture	133
Designing the Authority Escalation Path	133
The Production-Safe Prompt	134
Full Example: Authority Map and Status Library for the Lost-Phone Workflow	134
Common Designer Mistake: Authority Map as Afterthought	137
Design Practice: Authority Map for the Double-Charge Workflow	137
Key Takeaways	138
Opening Thesis	139
What Changes Over Time in an AI Workflow Environment	139
The Renewal Cadence Design	141
Incident Stewardship Protocol	142
Renewal Stewardship Protocol	143
The Steward’s Disposition	144
Monitoring Design	144
Full Example: One-Year Stewardship Cycle for the Bad-Restaurant-Experience Workflow	145
Common Designer Mistake: Confusing Monitoring with Stewardship	147
Design Practice: Stewardship Plan for the Founder-Offer Workflow	147
Key Takeaways	148
Reading These Cases	149
Overview	150
1. Messy Human Expression	150
2. Problem Statement	151
3. Solution Statement	151
4. Blind Ideal Simulation Summary	152
5. Pure To-Be Model	152
6. As-Is Discovery Questions	153
7. Likely As-Is Gaps	154
8. To-Be Selling Conversation	155
9. Authority Boundaries	156
10. Stewardship Risks	156
Overview	157
1. Messy Human Expression	157
2. Problem Statement	159
3. Solution Statement	159
4. Blind Ideal Simulation Summary	159
5. Pure To-Be Model	160
6. As-Is Discovery Questions	161
7. Likely As-Is Gaps	162
8. To-Be Selling Conversation	162
9. Authority Boundaries	163
10. Stewardship Risks	164
Overview	165
1. Messy Human Expression	165
2. Problem Statement	166
3. Solution Statement	167
4. Blind Ideal Simulation Summary	167
5. Pure To-Be Model	168
6. As-Is Discovery Questions	169

7. Likely As-Is Gaps	170
8. To-Be Selling Conversation	170
9. Authority Boundaries	171
10. Stewardship Risks	172
Overview	173
1. Messy Human Expression	173
2. Problem Statement	175
3. Solution Statement	175
4. Blind Ideal Simulation Summary	175
5. Pure To-Be Model	176
6. As-Is Discovery Questions	177
7. Likely As-Is Gaps	178
8. To-Be Selling Conversation	179
9. Authority Boundaries	180
10. Stewardship Risks	180
Overview	181
1. Messy Human Expression	182
2. Problem Statement	183
3. Solution Statement	183
4. Blind Ideal Simulation Summary	184
5. Pure To-Be Model	185
6. As-Is Discovery Questions	186
7. Likely As-Is Gaps	187
8. To-Be Selling Conversation	188
9. Authority Boundaries	189
10. Stewardship Risks	189
Working Instruments for the Practicing Designer	190
A Disciplined Memory	191
Template 1: Emergence Worksheet	192
Template 2: Problem Statement = Solution Statement Worksheet	194
Template 3: Blind Ideal Simulation Prompt Template	195
Template 4: Transcript Extraction Worksheet	197
Template 5: To-Be Model Template	198
Template 6: As-Is Discovery Interview Guide	200
Template 7: As-Is vs. To-Be Gap Table	205
Template 8: Authority Map Template	206
Template 9: Forbidden Claims Checklist	207
Template 10: Status Language Library	209
Template 11: Incident Stewardship Log	211
Template 12: Renewal Audit Template	212
Key Takeaways	215
Discomfort by Design	216
Lab 1: Emergence Hold	216
Lab 2: Blind Ideal Simulation	219
Lab 3: As-Is Discovery	221
Lab 4: False Completion Diagnosis	222
Lab 5: Stewardship Audit	223
Key Takeaways	224

Professional Discipline in Seven Gates	225
Checklist 1: Pre-Engagement	225
Checklist 2: Method 1 — Emergence Complete	227
Checklist 3: Method 2 — Blind Ideal Simulation Complete	228
Checklist 4: Method 3 — As-Is Discovery Complete	230
Checklist 5: Method 4 — To-Be Selling Complete	232
Checklist 6: Pre-Launch Production Safety	233
Checklist 7: Stewardship Review	235
Key Takeaways	237
What the Designer Carries Forward	237
Terms as Used in This Textbook	239
A Collected Reference	243

Preface

This book exists because a gap opened up.

On one side of the gap: a generation of practitioners who learned to design software by designing for compliance. Fill out the form. Choose the category. Follow the steps. Their craft was real — the discipline of forcing unruly human need into the clean grammar of a system. They were good at it. They built things that worked.

On the other side: AI that arrives with a different set of native capacities. AI that can hold a conversation without a script. AI that can tolerate ambiguity, follow a wandering sentence to its conclusion, and locate the resolved condition hiding inside a messy human expression. AI that, if you let it, does not need the form.

The gap is between those two realities — and almost no one is designing inside it deliberately.

This book is an attempt to do that.

What This Book Is Not

It is not another AI tools book. You will not find a ranked list of platforms, a comparison of large language models, or a guide to which tool to use for which task. Tools change. The principles governing how humans speak, how problems emerge, and how authority functions do not change at the same rate.

It is not a prompt engineering guide. Prompt engineering is a legitimate practice, and there are excellent resources dedicated to it. But prompt engineering is concerned primarily with what you put in — how to phrase an instruction to get a better output. This book is concerned with something different: the design of the full arc from a human speaking their first unfinished sentence to a problem genuinely resolved.

It is not a chatbot manual. Chatbots are artifacts. This book is about workflows — the full shaped structure of how an AI-assisted system moves a human from an unresolved state to a resolved one. The chatbot, if it appears at all, is one component in that structure, not the point of the exercise.

It is not, above all, an argument that AI will change everything or that organizations must act now or be left behind. Readers who want urgency rhetoric will find it easily elsewhere. This book assumes you are already here, already building or planning to build, and need something more useful than enthusiasm.

What This Book Is

This is a textbook in AI workflow design as a discipline. It teaches a specific framework — the five-method life cycle — for designing AI-assisted workflows from the moment a human begins speaking to the moment a problem is genuinely, verifiably resolved. It treats that arc seriously, attends to its phases carefully, and does not skip the hard parts.

The hard parts include: understanding what happens when a human speaks before they have fully formed what they mean. Understanding the difference between what AI can complete from pattern and what must be verified by authority. Understanding what false completion is and why it is a design failure, not merely a tone problem. Understanding how to use status language with precision. Understanding what the goal of a workflow actually is — not fewer clicks, not faster routing, not better categorization, but a human whose problem is genuinely no longer a problem.

Who This Book Is For

It is written for practitioners who design or build AI-assisted workflows: product designers, solution architects, AI implementation leads, operations designers, and technologists who are being asked to move their organizations from old-interface thinking into something new. It assumes comfort with professional design concepts but does not assume prior expertise in AI systems.

It is also written for the thoughtful generalist: the founder who is shaping an AI-assisted product and wants to understand the underlying discipline, the manager who is overseeing implementation and wants language for what is going wrong, the consultant who is being asked to evaluate an AI workflow and needs a principled basis for that evaluation.

How to Use This Book

The book is structured in five parts, corresponding to the five methods of the life cycle: Emergence, Hallucination, As-Is Discovery, To-Be Selling, and Stewardship. Each part builds on the last. The foundations in Part One — particularly the equation in Chapter Three and the distinction in Chapter Four — are load-bearing. Do not skip them.

Each chapter contains a main argument, at least one concrete example worked through in detail, a Common Designer Mistake section, and a Design Practice or Exercise section. The Design Practices are not optional enrichment. They are the point. Reading the argument without doing the practice is like reading a book about swimming.

A Note on the Recurring Examples

Throughout this book, you will encounter the same small cast of scenarios: a lost phone at a restaurant, a possible double charge on a credit card, a bad restaurant experience, a salesperson trying to reach a

decision maker, and a founder who cannot yet explain what their company does. These are not glamorous examples. They are not chosen to impress.

They are chosen because they are precise. Each one contains exactly the complexity that makes workflow design interesting — an unfinished human expression, an emotionally charged state, a gap between what the human knows and what the system needs, and a real authority structure that must be honored before resolution is genuine.

Simple service scenarios are the sharpest teaching instruments in this discipline. If you understand how to design a workflow for a customer who thinks they were double-charged but isn't sure, who called on a Saturday, who has the bank app open and is reading numbers aloud, who is scared and slightly embarrassed — if you can design that correctly, you can design almost anything.

That is what this book is for.

— John Rector # Introduction: Where Design Actually Begins

The Moment Before the Workflow

There is a moment that traditional workflow design ignores. It happens before the ticket is created, before the form is submitted, before the category is selected, before the routing rule fires. It is the moment when a human being opens their mouth and begins to speak.

That moment is where AI workflow design begins.

This is not a trivial reframing. It changes almost everything about how a workflow is structured, what its first phase must accomplish, how AI is deployed, and what success looks like. The moment a human begins speaking is the moment of emergence — and emergence, properly understood, is the foundational concept of this discipline.

Traditional workflow design begins too late. By the time a classical design process gets hold of the human, the human has already been converted. They have been converted from a speaking person with an unresolved problem into a user, a ticket, a category, a field set, a branch in a decision tree. The conversion happens through the interface: the intake form, the dropdown menu, the IVR phone tree, the chatbot that opens with “Please select from the following options.” Each of these mechanisms is an extraction device. They are designed to pull out specific pieces of structured information as efficiently as possible and route the result to the right place.

This is not wrong, exactly. It is just incomplete — and the incompleteness costs more than most organizations realize. What gets lost in the conversion is the emergence itself: the texture of what the human was actually experiencing, the full shape of the problem as they lived it, the details that didn't fit a category but that would have mattered. What gets lost, in short, is the raw material of a genuinely good resolution.

AI workflow design does not begin with the workflow. It begins with a human speaking.

The Central Thesis

The argument of this book can be stated plainly: the design of an AI-assisted workflow is the design of a full arc, from a human’s first unfinished expression to a problem’s genuine resolution. Every design decision along that arc must be made in the light of what is actually happening at each phase — not what is convenient to capture, not what is easiest to route, not what fits the data model.

What is actually happening at the beginning of that arc is emergence. A human being is speaking before they have fully formed what they mean. This is not pathology. It is the normal condition of live human speech. The designer’s first responsibility is to understand that condition and design in a way that honors it rather than collapsing it prematurely.

What is happening at the end of the arc is resolution — genuine, verifiable resolution. Not the feeling of resolution, not the completion of a process step, not the closing of a ticket. The designer’s final responsibility is to ensure that the system’s claim of resolution corresponds to an actual state in the world. This requires understanding the distinction between what AI can complete from pattern and what must be verified by authority — a distinction that this book treats with great care, because its violation is one of the most common and consequential failure modes in production AI systems.

Between beginning and end runs the full structure of the life cycle.

The Five-Method Life Cycle

This book is organized around a five-method life cycle that spans the full arc of AI-assisted workflow design. Each method names a distinct phase of that arc, with its own orientation, its own risks, and its own design requirements.

Method 1: Emergence. The phase in which a human speaks their problem aloud before it has fully resolved into a clean statement. Design here is concerned with holding the space for emergence rather than collapsing it into a category. The AI’s role is not extraction but presence — listening until the problem has fully arrived.

Method 2: Hallucination. The phase in which AI’s capacity for pattern-based completion is deployed deliberately and productively. In a discovery context, controlled hallucination — letting AI complete a pattern toward an ideal outcome — is a powerful tool for designing toward a future state. The critical skill is knowing when this capacity is a resource and when it becomes a liability.

Method 3: As-Is Discovery. The phase in which the current state of a workflow, system, or organization is mapped with precision. As-Is Discovery is not about what should be — it is about what is, in full and sometimes uncomfortable detail. The tools and disciplines of this phase borrow from process mapping, organizational analysis, and systems thinking.

Method 4: To-Be Selling. The phase in which a designed future state is articulated, tested, and communicated to the people who must choose to adopt it. This is not sales in the commercial sense. It is the practice of making a proposed resolution legible and compelling to those who must ratify or implement

it.

Method 5: Stewardship. The phase in which a deployed workflow is maintained, monitored, corrected, and renewed over time. Stewardship acknowledges that no workflow is permanently correct and that the ongoing relationship between the system and the humans it serves requires active tending.

These five methods are the spine of this book. Each part corresponds to one method. Within each part, the chapters develop the conceptual foundations, the design disciplines, and the practical skills required.

The Equation

Before entering Part One, it is worth introducing the central equation of this discipline, because it will appear in every chapter and every method.

Problem statement = solution statement.

This is not a metaphor. It is a structural claim about the relationship between an unresolved problem and its resolution. One side of the equation — the problem statement — may be messy, emotional, partial, repetitive, full of uncertainty. The other side — the solution statement — is clean and specific: it names the condition that would make the problem no longer a problem. Both sides are real. Both sides are valid. They are equivalent.

The equation matters because it gives the designer a target. Before routing, before classification, before any operational step, the designer must be able to name the solution statement. Without it, the workflow has no reliable destination. With it, every subsequent decision can be evaluated against the question: does this step move toward the named resolution?

The equation also matters because it changes how a designer listens to the problem statement. A messy problem statement is not a failed communication to be corrected. It is a valid expression of an unresolved condition — which means it contains, in rough form, everything needed to identify the resolution. The designer's job is not to clean up the problem statement. The designer's job is to identify its equivalent on the other side of the equation.

How This Book Is Organized

Part One: Foundations establishes the conceptual ground on which the entire discipline rests. Chapter One addresses the nature of human speech and what “arriving unfinished” actually means for design. Chapter Two examines the shift from interface design to translation design. Chapter Three develops the problem statement / solution statement equation in full, with worked examples across all five recurring scenarios. Chapter Four introduces the distinction between pattern and authority, defines false completion, and introduces status language as the production-safe solution to false completion.

Parts Two through Five address each of the remaining methods in the life cycle: Hallucination, As-Is Discovery, To-Be Selling, and Stewardship.

A note on sequence: the five methods are presented in order, but they are not strictly sequential in practice. Emergence recurs throughout the life cycle wherever a human speaks. Hallucination informs discovery. Stewardship feeds back into new emergence. The book's structure is a teaching sequence, not a claim that real design projects proceed in a straight line.

Begin with Chapter One. Begin with the human speaking. # Part One: Foundations

Before You Can Design Anything, You Must Know What You Are Designing For

There is a temptation, in any technical discipline, to begin with the technique. To reach for the tool, the framework, the model, the diagram. To show competence by producing an artifact quickly. This temptation is especially strong in AI workflow design, where the tools are visually impressive and the pressure to demonstrate capability is constant.

Part One resists that temptation entirely.

The four chapters that follow are foundational not in the sense that they are preliminary throat-clearing before the real content begins. They are foundational in the architectural sense: they bear the weight of everything built above them. Get these four concepts wrong — or skip them — and the workflows you design will be technically sophisticated and practically unreliable. They will look like progress and function like the old phone tree with a friendlier voice.

Chapter One begins with the human being who is speaking. Not the user. Not the customer. The human — arriving mid-thought, expressing something not yet fully formed, bringing language that is still becoming. Chapter One argues that this condition, which is universal, is the correct starting point for workflow design and that the field has been starting in the wrong place.

Chapter Two examines what it means to shift from building interfaces — systems that require humans to conform — to building translations — systems that meet humans where they are and carry their expression to resolution. This shift is not cosmetic. It changes the designer's fundamental orientation and demands a critical examination of instincts that were reliable in the previous paradigm.

Chapter Three develops the central equation of the discipline: problem statement equals solution statement. This equation is not a slogan. It is a working tool that gives every workflow a precise target and gives every designer a way to evaluate whether a given step is aimed at the right destination.

Chapter Four introduces the most consequential distinction in production AI systems: the difference between what AI completes from pattern and what must be verified by authority. It defines false completion — the specific failure mode that results from conflating these two capacities — and introduces the status language system that makes truthfulness operationally achievable.

These four chapters are the ground. Everything in this book is built on them. # Chapter 1: Humans Arrive Unfinished

Opening Thesis

Every speaking human arrives unfinished. This is not a statement about a particular type of user, a demographic, an education level, an emotional state, or a communication skill. It is a statement about the nature of live human speech itself. When a person opens their mouth to speak — unless they are reading from a prepared script, word for word — they do not know the exact next sound wave that will emerge. They know, roughly, what they mean. They may know the subject. They may feel the urgency. But the precise expression, the specific sentence, the exact word that will land next: that is still becoming.

This is the foundational fact of AI workflow design. Everything in the discipline flows from it.

The Sound Wave

Consider what actually happens when a human speaks. A sentence is not assembled in full before being transmitted. It is produced incrementally, through a process in which the act of speaking shapes what comes next. A speaker begins with something like a direction — an intention, a feeling, a rough destination — and generates sound waves that move toward it, adjusting with each word as the meaning clarifies. The speaker hears themselves. They revise. They add. They stop and restart. They say “I mean” and try again.

Linguists and cognitive scientists have studied this process extensively, and what they find is consistent: fluent human speech is not the output of a fully pre-formed plan. It is an online production, meaning it unfolds in real time with significant improvisation, revision, and emergence built into its structure. The pauses, the fillers, the restarts, the self-corrections — these are not failures of speech. They are the mechanism of speech. They are how meaning gets made under the pressure of real time.

This matters enormously for workflow design because most workflow design treats the human’s speech as if it were the transmission of a pre-formed message — as if the human arrived at the beginning of the interaction knowing exactly what they needed to say and merely needed to be guided through the correct fields for depositing it. This is wrong. Not slightly wrong. Structurally wrong. The human is not transmitting a pre-formed message. The human is generating meaning, in your presence, through the act of speaking.

The design implication is significant: if you interrupt the generation process — if you force a branch, demand a category, present a menu before the human has finished arriving — you are not helping the human communicate more clearly. You are stopping the generation before it has produced the full message. You are getting a partial expression and treating it as if it were complete. The workflow that follows from that partial expression is aimed at a target that has not yet fully appeared.

What “Unfinished” Means

When this book says that humans arrive unfinished, it means something precise. It does not mean confused. It does not mean unintelligent. It does not mean poorly prepared or emotionally dysregulated,

though any of those may also be true. It means that the expression of the problem is still becoming — that the full shape of what the human needs to communicate has not yet arrived in language.

A person calling a restaurant to report a lost phone knows, with perfect clarity, that they have lost a phone. They know where they think they left it. They know how urgent it is. What they do not yet have is the organized, complete, all-necessary-details statement of the situation that the system will eventually need to act on it. They have a feeling and a direction. The full statement is ahead of them, and they will reach it through the act of speaking — if they are allowed to.

A person calling their credit card company about a possible double charge knows they are worried. They may not know yet whether it was actually a double charge or a recurring subscription they forgot about. They have the bank app open. They are reading numbers aloud as they look. Their problem statement is not yet complete because their understanding of the situation is not yet complete. The act of speaking, in this case, is also the act of thinking through the problem — and the thinking is not done when the first sentence arrives.

A person calling a restaurant the day after a bad experience knows they had a bad experience, but what they want is less clear. They may want acknowledgment. They may want a refund. They may want someone to know that the entrée was cold. They may not want anyone to get in trouble. All of this is in motion as they speak. The expression of it will take time to arrive. Cutting that process short — routing them to a complaint form, asking them to press 1 for service issues — is a category error that mistakes the beginning of emergence for the complete statement of a problem.

In each case, “unfinished” names the gap between what the human knows and what the human has yet fully expressed. That gap is not a failure. It is the work of the conversation. The designer’s responsibility is to ensure the system is built to hold that gap open long enough for the full expression to emerge.

Why Old Workflow Design Begins Too Late

Classical workflow design begins with the assumption that the human’s need can be captured, classified, and routed. The intake form, the dropdown menu, the IVR phone tree, the categorized support ticket — all of these are mechanisms for capture. Their logic is: get the human’s need into a structured form as quickly as possible, so that it can be handled by the right part of the system.

This logic has genuine merit in a world where the system cannot hold a conversation. If your system can only execute on structured inputs, then your only option is to force humans to produce structured inputs. The intake form is not perverse — it is the best available solution to the problem of translating unstructured human need into a form the system can act on.

But AI changes this. AI can hold a conversation. AI can follow an unfinished sentence to its conclusion. AI can tolerate ambiguity, reflect back what it has heard, ask a clarifying question that does not feel like a demand, and wait while the human thinks. AI does not need the intake form. And yet — and this is the central irony of the current moment — most AI workflow design is built on the same logic as the intake form, just with a friendlier voice.

The AI workflow asks: “What brings you in today? Is this a billing issue, a technical issue, or something else?” This is a dropdown menu spoken aloud. It has the same defect as the dropdown: it asks the human to select a category before the full problem has emerged. If the human’s problem is a billing issue, they may accurately say “billing issue” — but if their problem is a billing issue that is actually a symptom of a technical issue, or a billing issue that they are not sure is really an issue, or a billing issue that they feel embarrassed to discuss, the category collapses the complexity before it has been heard.

Old workflow design begins too late because it begins at the moment of classification rather than the moment of emergence. It treats the first structured output — the category, the ticket type, the routed destination — as the beginning of the work, when in fact the beginning of the work is what happened before that: the human speaking, the problem taking shape, the full expression arriving.

The beginning of the workflow is not extraction. The beginning of the workflow is emergence.

The Golf Analogy: Do Not Help the Ball

In golf instruction, there is a persistent problem among beginners. When the ball is not going where they want it to go, beginners instinctively try to help the ball. They tilt their body in the direction they want the ball to go. They scoop with the club rather than strike. They intervene at the moment of impact in ways that feel helpful and are reliably destructive. The well-intentioned correction undermines the very mechanics that would have produced the desired result.

A good golf instructor teaches the student to stop helping the ball. The ball does not need help. The physics of a properly executed swing will carry the ball where it needs to go. The student’s job is to execute the swing correctly and trust the result. Intervening at impact, however intuitive it feels, is the problem, not the solution.

The analogy to AI workflow design is direct. AI already knows how to have a conversation. It does not need you to make it conversational. It has been trained on more conversations than any human being will ever have. It knows how to follow an unclear sentence. It knows how to ask a gentle question. It knows how to reflect back what it has heard. It knows how to wait.

The designer’s job, during the emergence phase, is not to make AI conversational. The designer’s job is to not un-make it conversational. Do not help the ball.

The specific failure mode — helping the ball — looks like this: the designer adds intake questions, early routing prompts, category confirmation requests, structured data capture sequences, all in the sincere belief that these additions will make the AI more efficient and the user experience smoother. In fact, these additions interrupt the emergence process and revert the system to old-interface behavior. The AI asks the questions the designer put in front of it. The human conforms to the structure the designer imposed. The problem is partially captured. The workflow is aimed at a partial target.

The discipline required here is restraint. Knowing that you have the capability to add structure, and choosing not to add it during the emergence phase, because the emergence process is doing something more important than any structure you could impose.

What the Designer’s Job Actually Is

If the designer’s job during emergence is not to add structure, what is it?

It is to design the container for the emergence — to ensure that the AI’s native conversational capacity is deployed in a context that allows the full problem to arrive. This means several things in practice.

It means the designer must understand what the emergence phase is, what it produces, and how it transitions into the next phase. Emergence is not an indefinite open-ended conversation. It has a shape. It begins when the human starts speaking and ends when the full problem statement has arrived — when the human has expressed, in enough completeness, the situation they are in and what they need. The designer’s job is to understand that shape and to know when emergence is complete.

It means the designer must resist the impulse to rush the transition. Emergence ends when it ends, not when it would be convenient for it to end. A customer calling about a bad restaurant experience may need several exchanges before the full shape of what they want becomes clear. An AI that routes to “complaint resolution” after the first sentence — “I had a bad experience last night” — may be routing to the right department but has not yet heard the problem. The problem includes the cold entrée, the server who was actually excellent, the concern about not getting anyone in trouble, the question of whether they should have said something at the time, and the emotional residue of spending a significant amount of money on a disappointing evening. All of that is the problem. The first sentence was the beginning of emergence.

It also means the designer must know what the emergence phase is producing. It is producing the problem statement — the full, messy, human expression of an unresolved condition. That problem statement is the input to everything that follows. It is the material from which the solution statement will be identified. Getting it right — getting all of it — is the work of emergence.

The Temptation to Skip Emergence

There is a reason designers skip or compress the emergence phase. It feels inefficient. It does not produce structured data. It is harder to measure than a form submission or a category selection. It is harder to hand off to a downstream system. It does not fit neatly into the existing data model.

These are real pressures, and they are not unreasonable. But they are pressures that, when acted on prematurely, produce workflows that serve the system rather than the human. The efficiency of capture is not the same as the effectiveness of resolution. A workflow that captures a partial problem statement very efficiently and routes it accurately is not a good workflow. It is a fast workflow aimed at the wrong target.

The case for protecting the emergence phase is ultimately a case about downstream quality. A complete problem statement produces a more accurate solution statement, which produces a more precisely designed workflow, which produces a genuine resolution. The investment in emergence is returned many times over in the downstream phases — and the cost of skipping it is not paid in the emergence phase, where nothing looks wrong, but in the resolution phase, where the human is handed an answer to a question they did not quite ask.

Common Designer Mistake

Adding Intake Structure to the Emergence Phase

The most consistent mistake designers make in this phase is importing old-interface logic into the new design context. It manifests as a set of very reasonable-sounding additions: an intake question at the top of the conversation, a category confirmation prompt, an early routing decision to “get the customer to the right place faster,” a structured field capture sequence disguised as natural language.

Each of these additions feels like an improvement. The designer has reduced ambiguity. The system knows earlier what it is dealing with. The routing is cleaner.

What has actually happened is that the emergence phase has been collapsed before the problem fully arrived. The human was asked to select a lane before they finished speaking. The problem that arrived in that lane is a partial problem — the part that fit the structure offered, stripped of the parts that did not.

This mistake is particularly seductive in AI design because the language can be made to sound like a conversation. “I want to make sure I get you to the right place — is this more of a billing question or a service question?” sounds warm and helpful. It is, structurally, a dropdown menu. It forecloses the parts of the problem that live in the space between billing and service, or that are billing and service and something else the customer has not yet named.

The designer who catches themselves adding structure to the opening of the workflow should ask: am I doing this because the emergence is complete, or am I doing this because I am uncomfortable with how long emergence takes? The answer is almost always the latter.

Design Practice

Observing the Moment of Collapse

Select a real service interaction — a recording, a transcript, a customer service call you have direct access to, or a publicly available example of a customer service conversation. It does not need to be an AI interaction. A human-to-human service call is equally instructive, sometimes more so.

Read or listen to the interaction from the beginning. Your task is to identify the exact moment at which the human was first collapsed into a category.

Collapse happens when the system — human or AI — makes a move that asks the human to stop emerging and start conforming. It might be a direct question (“Is this about your account or a recent order?”). It might be a form field offered mid-conversation. It might be a routing statement (“Let me transfer you to our billing department”). It might be an affirmation that simultaneously closes down: “Got it — so this is a billing issue.”

Mark that moment precisely. Then ask: what was the human still in the process of expressing at that moment? What had not yet arrived in the conversation? What details, qualifications, emotional context, or secondary concerns were still becoming when the collapse occurred?

Write a short paragraph — no more than 200 words — describing what was lost. Not what might have been lost, not what the human should have said, but what the evidence in the transcript suggests was still in process when the collapse happened.

This practice develops the designer's ear for emergence and trains the specific skill of recognizing the collapse before it happens — which is the prerequisite for designing a system that does not cause it.

Key Takeaways

Every speaking human arrives unfinished. This is not a user type or a problem to be corrected. It is the structural condition of live human speech, and it is the correct starting point for AI workflow design.

The emergence phase is the phase in which the full problem statement arrives. It begins when the human starts speaking and ends when the problem has been fully expressed — which takes longer than most intake processes allow.

Old workflow design begins too late because it begins with classification rather than emergence. The act of classification — dropdown, category, routing prompt, field set — collapses the emergence phase before the full problem has arrived.

AI already knows how to hold a conversation. The designer's job is not to make AI conversational. The designer's job is to not un-make it conversational by imposing structure during the emergence phase. Do not help the ball.

The beginning of the workflow is not extraction. The beginning of the workflow is emergence.

The efficiency of capture is not the same as the effectiveness of resolution. A workflow that captures a partial problem statement very efficiently is not a good workflow. It is a fast workflow aimed at the wrong target. # Chapter 2: From Interface to Translation

Opening Thesis

For most of the history of software, humans had to speak computer. They learned the grammar of the system: the correct menu path, the acceptable field format, the required sequence of steps. Skill with software meant skill at conforming to the system's expectations. The better you were at conforming, the more the system worked for you. The worse you were at conforming — or the more your need failed to fit the system's categories — the less the system could help you.

AI changes this relationship. It does not change it entirely or automatically, and it does not change it without intentional design. But AI introduces, for the first time in the history of digital systems, the genuine possibility that a human being can arrive without conforming — can speak naturally, incompletely, in their own vocabulary, in their own emotional register — and still be met.

This shift demands a new design orientation. Not an updated toolkit. Not a modified interface. A different fundamental stance. The designer must move from designing for user compliance to designing for emergence and resolution. This chapter names that move, examines its implications, and addresses

the most significant obstacle to making it: the instincts formed by years of doing the previous thing very well.

What an Interface Is

An interface, in the traditional sense, is a constraint layer. It is the designed surface between a human being and a system — and its defining characteristic is that it places the burden of translation on the human. To use the interface correctly, the human must understand the system’s grammar well enough to express their need in a form the system can process.

This is not incidental. It is the point. An interface designer’s job, in the classical formulation, is to make the constraint layer as learnable, as navigable, as forgiving of human error as possible — while still ensuring that the output conforms to what the system requires. The art of interface design is in reducing friction for the human on the way to compliance. The best interfaces make compliance feel easy, even enjoyable. But the destination is always compliance.

Consider what this means in concrete terms. A customer who wants to dispute a charge on their credit card does not simply say, “I think I was charged twice for something in March.” They navigate to a dispute screen, select the transaction from a list formatted in the system’s notation, choose a reason code from a predetermined list, type a description into a character-limited field, and submit. If they select the wrong reason code, the dispute may be routed incorrectly. If their description does not match the expected format, it may be flagged for manual review. The system is not hostile. It is simply designed to receive structured input, and the human must produce structured input to use it.

The burden is on the human. This is what an interface is.

What Translation Means

Translation, in the sense used throughout this book, is something different in kind. It is the capacity to stand between an unfinished human expression and a resolved operational outcome — and to carry the expression across to resolution without requiring the human to produce structured input first.

This is what AI, designed well, can do. AI can receive the unstructured expression — “I think I was charged twice for something in March, I have the app open, it shows two things on the fourteenth, one of them looks like it might be a subscription but I’m not sure, I never set up a subscription, I don’t know if this is a me problem or a bank problem” — and carry it forward. Not by classifying it immediately. Not by routing it to the right department. By staying with the human in the emergence, asking what is needed, and gradually producing, through the conversation, the structured understanding that the downstream system will eventually need.

AI is a translator, a living bridge, a pattern-bearing participant. This is a different design concept than interface. An interface is static: it presents its constraints and waits for the human to meet them. A translator is active: it meets the human in their expression and works toward a shared understanding.

The translator does not demand that the human speak the system's language. The translator does the work of moving between the human's language and the system's language — in both directions.

This capacity is real. It is not hypothetical or aspirational. It exists in contemporary AI systems right now, and it is routinely underused because the workflows built around those systems are designed with interface logic rather than translation logic.

The Discipline Shift

Moving from interface design to translation design requires a shift in every major design assumption.

In interface design, the primary question is: how do I get the human's need into the correct format as efficiently as possible? The design answer is constraint: menus, fields, dropdowns, validation rules, step sequences. Each constraint reduces the range of possible inputs, which makes the input more processable and the output more reliable. The tradeoff — forcing humans to conform — is accepted as a necessary cost.

In translation design, the primary question is: how do I allow the human's need to arrive in full, and carry it to resolution? The design answer is space: holding the emergence phase open, allowing the problem statement to develop, asking questions that widen rather than narrow, and delaying structural capture until the full problem has arrived.

This is not an argument for removing all structure from AI workflows. Structure is necessary. The downstream system still needs processable input. Authority still requires specific information. Routing still happens. The argument is about when structure is imposed — and the answer that translation design gives is: later. After emergence. Not at the beginning.

The designer trained in interface thinking will feel, viscerally, that this is wrong. It feels inefficient. It feels like it will produce a mess. It feels like allowing the human to “just talk” will result in a conversation that never resolves. These feelings are understandable. They are the correct feelings for the previous paradigm. They are unreliable guides in the new one.

The Risk of Interface Thinking in AI Contexts

Interface thinking, imported wholesale into AI design, produces a specific and consistent failure mode. It produces the AI workflow that replicates the structure of the IVR phone tree — the spoken dropdown menu — while adding the warmth of natural language. The AI asks early routing questions. It confirms categories quickly. It moves into structured data capture as soon as possible. It behaves, functionally, like the old system with a friendlier voice.

This failure mode is common enough that it deserves a name: premature structuring. Premature structuring happens when the designer imposes organizational structure on the conversation before the human's expression has arrived in full. The designer intends to help. The result is that the AI collapses the emergence phase and becomes, in effect, an expensive IVR system.

The human on the other side of a premature-structuring AI workflow may not even notice what was lost. The conversation felt organized. The AI seemed to understand quickly. The routing was smooth. What the human may notice later — if they notice it at all — is that the resolution did not quite fit. The answer was good but not exactly right. The process was completed but something felt unaddressed. This is the downstream cost of premature structuring: a resolution aimed at a partial target.

Consider the founder who has a new business idea but cannot yet explain what it does. This is a real and common situation. The idea is in motion; the vocabulary for it has not yet crystallized. An interface-thinking AI will offer categories — “Is this a product or a service? What industry?” — and the founder will select the closest available answer. The AI will proceed with a workflow optimized for that category. The result will be a perfectly executed workflow aimed at a business that is not quite the one the founder has in mind. The categories foreclosed the parts of the idea that did not fit any of them — which, for a genuinely new idea, may be the most important parts.

A translation-thinking AI will hold the emergence open. It will follow the founder’s partial, exploratory expression as far as that expression goes, reflect back what it has heard, ask questions that invite more rather than less, and wait while the idea arrives more fully. The process takes longer. The result is a solution statement that actually corresponds to the founder’s situation.

AI Does Not Need You to Make It Conversational

This point deserves direct statement, because it is often misunderstood.

When organizations undertake an AI workflow project, they frequently identify “making the AI conversational” as a design goal. The AI should feel natural. It should feel warm. It should not feel like a robot. These are reasonable concerns, and the impulse behind them is correct. But the implementation that follows from this framing is usually wrong.

Making AI conversational, in practice, often means adding scripted warmth to a fundamentally non-conversational structure. The AI says “Great question!” before routing the human into a predetermined branch. It says “I understand how frustrating that must be” before presenting a categorization menu. The conversational elements are decorative. The underlying structure remains an interface.

This is the wrong diagnosis. The problem is not that AI sounds too robotic. The problem is that the workflow is too rigid. A genuinely conversational AI workflow is one that allows the emergence phase to function — that lets the human speak, follows where the speaking goes, and holds open the space for the problem to arrive. This is not achieved by adding warm language to a rigid structure. It is achieved by removing rigid structure from the emergence phase.

AI does not need you to make it conversational. It needs you to not un-make it conversational. The training that produced contemporary AI has given it more conversational capacity than it is typically permitted to use in production workflows. The design move is not to add conversation. The design move is to stop subtracting it.

Leaning into the nature of AI means trusting the native conversational capacity that is already there and

building workflows that use it rather than suppress it. It means recognizing that the AI’s ability to tolerate ambiguity, follow partial expressions, and ask questions that open rather than close is not a liability to be managed. It is a resource to be deployed.

What “Leaning Into the Nature of AI” Means in Practice

This phrase will recur throughout the book, and it is worth giving it a precise definition here. Leaning into the nature of AI, in the context of workflow design, means designing in alignment with what AI actually is — a pattern-bearing participant in conversation that can hold ambiguity, complete partial expressions, and work across registers of formality and emotion — rather than designing as if AI were a faster, friendlier version of the systems that came before it.

In the emergence phase, it means protecting the space for unfinished speech. It means not adding routing prompts at the beginning. It means allowing the human to express the problem in their own words, for as long as that takes, before moving into any structural phase.

In the translation phase — the movement from problem statement to solution statement — it means using AI’s pattern capacity to identify what the clean resolution looks like, without asserting that the resolution has occurred. (This distinction, between pattern-based completion and authority-verified completion, will be developed fully in Chapter Four.)

In the operational phases — routing, data capture, authority verification, status tracking — it means using AI’s language capacity to make every step in the process legible and human to the person experiencing it, without allowing that language capacity to generate false reassurance.

Leaning into the nature of AI is not about enthusiasm for the technology. It is about accurate design — about building systems that use what is actually there rather than what was there before.

Common Designer Mistake

Building a Spoken IVR

The most consistent mistake in AI workflow design is replicating the IVR phone tree structure in spoken form. The IVR was a product of its constraints: a telephone system that could respond to button presses and play recorded messages. Its logic was branching: the human pressed 1, which triggered a response tree, which offered further presses, which eventually reached a destination. The IVR asked the human to speak the system’s language, literally — press 1 for billing, press 2 for technical support.

AI has none of these constraints. It does not require button presses. It does not require the human to select from a predetermined list. It does not require the problem to fit a predetermined branch. And yet the single most common AI workflow design pattern in production today is: AI asks an early question that functions as a branch selector, receives an answer, and proceeds down the selected branch.

“What brings you in today — is this about a billing issue, a technical issue, or something else?” is “Press 1 for billing, press 2 for technical support, press 0 for other” with a conversational wrapper. It has

the same structural flaw: it asks the human to self-categorize before the problem has arrived, it presents categories that may not contain the problem, and it offers an “other” escape hatch that catches everything the categories missed — which is often everything important.

The designer who has built IVR systems, web intake forms, or tiered support routing trees will find this pattern feel deeply natural. It is familiar. It is measurable. It can be diagrammed. It feels like a workflow.

It is also wrong. It is interface logic applied to a translation context. The discipline required to avoid it is the discipline of recognizing, every time you reach for an early routing question, that you may be building a spoken IVR — and asking whether that is really what the situation calls for.

Design Practice

Redesigning the First Sixty Seconds

Select an existing IVR or web form intake workflow. It can be from your organization or a publicly available example. The more familiar you are with it, the more instructive the exercise will be.

Step one: map the first sixty seconds of the workflow. What does the system do? What does it ask? What does the human produce in response? What gets captured? What gets routed? Do this mapping in enough detail that you can see exactly where the human is asked to conform — the first moment the system presents a constraint.

Step two: for each constraint you have mapped, ask what the constraint is protecting against. What failure mode does this constraint prevent? What does the system need that the constraint is designed to produce? Write a sentence for each constraint describing the underlying need it serves.

Step three: redesign the first sixty seconds as an emergence phase. Remove the constraints. Replace the routing questions, category confirmations, and structured capture prompts with an open invitation for the human to speak. What does the AI say instead? Write the opening lines of the new workflow — the invitation, and the first two or three AI responses that hold the emergence open.

Step four: identify what is different. What changed between the original first sixty seconds and the redesigned version? What will the system know at the end of the original sixty seconds versus the end of the redesigned sixty seconds? Which version knows more? Which version has more accurately captured the problem? Which version has a cleaner target for the resolution that follows?

This exercise develops the practical skill of translation thinking — the capacity to see where interface logic has been applied and to replace it with a design that holds the space for emergence.

Key Takeaways

An interface places the burden of translation on the human. The human must speak the system’s language — select the correct category, fill in the correct field, follow the correct sequence — to receive service.

AI changes this relationship. AI can stand between an unfinished human expression and a resolved operational outcome. AI is a translator, a living bridge, a pattern-bearing participant. It is not a smarter interface. It is a different kind of thing.

The shift from interface design to translation design is a shift in every major design assumption: from efficiency of capture to completeness of emergence, from constraint as the primary tool to space as the primary tool, from structuring early to structuring after the problem has arrived.

Interface thinking, imported into AI design, produces premature structuring — the imposition of organizational structure before the human’s expression has arrived in full. The most consistent manifestation is the spoken IVR: a rigid branching structure wrapped in natural language.

AI does not need you to make it conversational. It needs you to not un-make it conversational. The design move is to stop subtracting from AI’s native capacity, not to add warmth to a rigid structure.

Leaning into the nature of AI means designing in alignment with what AI actually is — trusting its native capacities and building workflows that use them rather than suppress them. # Chapter 3: Problem Statement Equals Solution Statement

Opening Thesis

The most important equation in AI workflow design is not a formula for routing or classification. It is not an algorithm, a scoring function, or an optimization target. It is a structural claim about the relationship between an unresolved problem and its resolution, and it can be written simply:

problem statement = solution statement

This equation is the central design tool of Method 1. It gives every workflow a precise target. It gives every designer a way to evaluate whether a given step is moving toward the right destination. And it provides — when properly understood — a complete corrective to the most common structural error in AI workflow design: building a workflow aimed at the wrong thing.

This chapter develops the equation in full. It explains what each side means, why the equation holds despite apparent asymmetry between its sides, how to use it in practice, and — critically — what Method 1 actually is and is not. Method 1 is only about this equation. Understanding that boundary is as important as understanding the equation itself.

The Equation

Begin with what the equation does not say. It does not say that problem statements and solution statements look the same. They almost never do. It does not say that you can derive one mechanically from the other by substituting terms. It does not say that a messy problem statement automatically produces a clean solution statement by some automatic process.

What it says is that they are equivalent. They refer to the same situation from two different positions: the unresolved position and the resolved position. The problem statement describes the situation as it

currently exists — with all its mess, uncertainty, emotional freight, and incompleteness. The solution statement describes the condition that would make the problem no longer a problem — with the clarity and specificity that resolution allows.

Think of an algebraic equation in which one side is complex and unsimplified — full of parentheses, fractions, unknown terms — and the other side is a clean integer. The complex side and the clean side are not different in value. They are different in form. The work of the equation is the work of showing that the complexity on the left is equivalent to the clarity on the right. Nothing is added or removed. A different form of the same reality is revealed.

This is exactly the relationship between the problem statement and the solution statement. The problem statement is the complex, unsimplified side: messy, partial, emotionally charged, full of uncertain terms, repetitive in places, incomplete in others. The solution statement is the simplified side: clean, specific, naming the resolved condition. They are not different in substance. They are the same situation expressed in two different forms.

Why “Messy” Is Not “Wrong”

This point bears direct emphasis because the instinct to correct messy problem statements — to clean them up, standardize them, translate them into a proper format — is strong and consistently misdirected.

A messy problem statement is the problem as live human speech expresses it. It contains everything that live human speech contains: uncertainty, emotion, repetition, missing details, undefined terms, side comments, contradictions, revisions, and tangents. None of that makes it incorrect. All of it makes it real.

The messiness is not noise. The messiness is signal. The uncertainty the human expresses is real uncertainty — about whether the charge was really a double charge, about whether the food was really that bad or whether they are just tired, about whether they want a refund or just acknowledgment. The emotion is real emotion. The side comments often contain the most important information: the detail that doesn't fit the category, the qualification that changes everything, the thing the human said almost by accident that reveals the actual shape of the problem.

When a designer instinctively tries to clean up the problem statement — standardize the terminology, remove the emotional content, strip out the side comments — they are reducing the signal. They are producing a tidy version of the problem that is easier to route but less accurate as a description of what the human actually needs. The solution statement that follows from the tidy version will be cleaner and less accurate than the solution statement that would have followed from the messy original.

The discipline is to resist the cleanup impulse and to read the messiness carefully instead. What is the uncertainty actually about? What is the emotion telling you about the stakes? What is the side comment revealing about the secondary concern? These questions produce a richer solution statement — and a better resolution — than the cleanup alternative.

The Clean Solution Statement

The solution statement does not need to be complex. In fact, it should not be. Its job is to name, with precision, the condition that would make the problem no longer a problem. Nothing more, nothing less.

The solution statement is not a plan. It is not a workflow. It is not a set of steps. It is not a department assignment or a routing destination. It is a condition — a state of affairs that, if it existed, would mean the problem was resolved.

This is a simple concept and a difficult discipline. The difficulty is that once the problem has been heard, the designer's instinct is to jump immediately to the plan: what steps need to happen, in what order, to achieve the resolution? This jump is premature. Before the plan can be designed, the destination must be named. The solution statement is the destination.

If the destination is named imprecisely — or not named at all, because the designer moved directly from problem statement to routing logic — the plan that follows will be aimed at a fuzzy target. The steps may be executed correctly. The routing may be perfect. The workflow may complete without error. And the human, at the end of it, may find that the resolution doesn't quite fit — because the destination was wrong, or vague, or only partially identified.

The Five Recurring Examples

The best way to develop skill with the equation is to practice it repeatedly across varied scenarios. The five recurring examples in this book are designed for exactly this purpose. Each one represents a different configuration of mess, emotion, and uncertainty in the problem statement — and a correspondingly different kind of clarity in the solution statement.

The Lost Phone

Problem statement (as it arrives): A person calls a restaurant. They were there last night. They think they left their phone. They are not sure — maybe it fell out in the car, but they checked and didn't see it. It's a black phone, there might be a case on it, actually they're pretty sure there's a case, it's like a clear case maybe? They've been calling it from their partner's phone and it just rings, so it might be on silent. They're supposed to have it for a call this afternoon and they're a little panicked.

The messiness is real: uncertainty about whether the phone is at the restaurant at all, uncertainty about the physical description, the detail about calling from a partner's phone, the mention of silent mode, the time pressure, the emotional state.

Solution statement: The person has their phone in their possession.

That is the entire solution statement. Not: the restaurant conducts an inventory of found items. Not: a lost-and-found report is filed. Not: the phone is identified among found items. Those are steps in a workflow. The solution statement is the condition — the person has their phone. Everything else is in service of that.

The Possible Double Charge

Problem statement (as it arrives): A person calls about their credit card. They were charged twice, they think — they're looking at their statement right now, actually at the app, and there are two transactions on the fourteenth, both for the same amount, thirty-seven dollars and something, one says it's from a coffee place they go to and the other one just says the same name but slightly different. They don't know if that's how it works, like maybe there's a hold and then the actual charge? They've never had this happen. They're worried because they're watching their budget this month and they just want to make sure everything is right. They're not trying to make trouble, they just want to know.

The messiness is dense: uncertainty about whether it's really a double charge or a hold-then-charge sequence, the detail about two slightly different transaction names, the emotional content around budget anxiety, the preemptive self-diminishment (“not trying to make trouble”), the real-time reading of the app.

Solution statement: The person's financial situation accurately reflects what they actually spent.

This solution statement is deceptively broad, and that breadth is intentional. It accommodates both possible outcomes: if the charge was legitimate, the solution is that they understand why and feel confident about their account. If the charge was an error, the solution is that it is corrected. In either case, the same condition applies: their financial situation accurately reflects reality.

The Bad Restaurant Experience

Problem statement (as it arrives): A person calls a restaurant the day after their dinner. They say they had a bad experience. When asked what happened, they are somewhat uncertain: the food took a long time, and then when it came, their entrée was cold. But actually the server was really nice, they don't want anyone to get in trouble. They had the salmon. They debated saying something at the time but they didn't want to make a scene. They paid the full bill. They're a regular — they've been coming for years — and they just feel like it wasn't up to the restaurant's usual standard. They're not sure what they're looking for, really.

The messiness is rich: qualified complaint (it was bad, but the server was nice), conflicting impulses (want acknowledgment but don't want anyone to get in trouble), secondary concern (they're a regular, this is about the relationship), expressed uncertainty about what they want, and an unresolved question about billing that is implied but not asked directly.

Solution statement: The person's experience is made right in a way that honors both the complaint and the relationship.

The solution statement must capture both dimensions — the legitimate grievance and the relational context — because the problem statement contains both. A solution statement that names only the grievance (“refund issued”) would miss the relationship. A solution statement that names only the relationship (“customer retention achieved”) would miss the grievance. The full solution statement must be equivalent to the full problem statement.

The Salesperson and the Decision Maker

Problem statement (as it arrives): A salesperson calls and reaches the front desk. They say they are trying to get in touch with the right person about something that could be a good fit for the company. They don't know who specifically handles this kind of thing. When asked what it is, they say it is a solution for streamlining some operational processes — they'd rather explain it to the right person than go through the whole thing multiple times. They are friendly and professional but not quite forthcoming about specifics. They mention they've worked with similar companies.

The messiness is strategic: the salesperson's problem statement is deliberately incomplete, designed to get a name without committing to a pitch before reaching the decision maker. The actual problem, seen from the outside, is that the salesperson does not have access to the person who can say yes.

Solution statement: The salesperson is connected to the person who has authority to evaluate and act on the offer.

This solution statement is useful because it reveals what the routing challenge actually is. The question is not “what department handles this?” The question is “who has authority?” Those are often different answers, and a workflow that routes to a department rather than to an authority may route correctly and still not produce the resolution.

The Founder with the Unclear Offer

Problem statement (as it arrives): A founder is trying to explain what their company does. It has been described different ways at different times. Sometimes they say it is a platform. Sometimes they say it is a service. It helps companies — mostly mid-size companies — with the kind of problems that don't fit in any one department. It is kind of like consulting but with software. Or maybe software with a consulting component. They've been working on the pitch and every version seems to either be too broad or too specific. They know what it does when they are in a room with the right person, but they can't seem to get it onto paper in a way that works.

The messiness is conceptual: the idea exists but has not yet been made communicable. The founder knows the thing but cannot yet say the thing.

Solution statement: The offer is made communicable in a form that works consistently outside the founder's presence.

The solution is not “the pitch is written.” The pitch is a step in a workflow. The solution is the condition: the idea can be communicated reliably, in the founder's absence, to the right audience.

What Method 1 Is, and Is Not

These five examples illustrate the work of Method 1 precisely: hear the problem statement, identify the equivalent solution statement, name the resolved condition. That is all.

Method 1 is not about routing. Routing is a downstream activity that happens in a later phase, once the

solution statement has been named and the workflow can be aimed at the right target. Method 1 is not about fields or databases or APIs or records or departments or forms. Those are operational concerns that belong to the phases that follow. Method 1 is not about designing anything. It is not a workflow. It does not have steps or branches or decision points.

Method 1 ends when two things are true: the problem statement has been fully heard, and the equivalent solution statement has been named. That is the output of Method 1. One sentence, or perhaps two, that names the condition that would make the problem no longer a problem. Everything that comes after is built on top of that named condition.

The discipline of keeping Method 1 bounded — of not letting it sprawl into routing, field design, or operational planning — is a real discipline. The impulse to move from the solution statement directly into the plan is strong. It feels like progress. It feels like efficiency. It is, in fact, a jump over the most important step in the workflow: the moment when the designer confirms that the target has been correctly identified before building the machinery aimed at it.

“The goal is not fewer words. The goal is fewer wrong workflows.” A workflow built without a clearly named solution statement is not a lean workflow. It is a workflow aimed at a vague target — and it may be executed with great precision in entirely the wrong direction.

Common Designer Mistake

Jumping from Problem Statement to Routing Logic

The most consistent mistake in the context of this equation is moving from the problem statement directly to routing logic, operational steps, or field design before the solution statement has been clearly named.

This mistake has a specific signature: the designer can describe what will happen in the workflow — the steps, the routing, the system updates, the notifications — but cannot complete the sentence “the problem is resolved when...” without consulting the workflow description.

The consequence of this mistake is subtle but consequential. The workflow is built with a destination that was implied rather than named. If the implied destination is correct — if the routing logic happens to point at the right resolution — the workflow functions. But if the implied destination is slightly wrong — if there was a secondary concern in the problem statement that was not captured in the routing logic, or if the human’s actual resolution condition was different from what the routing assumed — the workflow executes correctly and produces a result that doesn’t quite fit.

The designer who made this mistake has no way of knowing, from the workflow itself, whether the resolution is correct. They did not name the destination precisely enough to evaluate whether it was reached. They know the steps were completed. They do not know if the right thing was accomplished.

The correction is deliberate: before touching routing, fields, or operational design, write the solution statement. Write it as a condition, not a process. Write it in language that would be meaningful to the human who stated the problem — not in system language, not in department language, not in field names.

Then evaluate the routing logic against the condition. Does this routing, if executed correctly, produce the named condition? If yes, proceed. If not, revise the routing before building it.

Design Practice

Naming the Equivalent Resolved Condition

Below are five problem statements written as realistic spoken transcripts — the kind of thing that actually arrives in the first few minutes of a service conversation. For each one, your task is to write the solution statement using the equation. Do not route. Do not classify. Do not design a workflow. Just name the equivalent resolved condition — the state of affairs that would make this problem no longer a problem.

Problem Statement 1: “Hi, yes, I was at your restaurant on Friday — I think it was Friday, maybe Thursday, we’ve been really busy — and I’m pretty sure I left my phone there. It’s a newer model, kind of big, I had it at the table, I was showing my wife something on it and then I think I just left it when we got up because we were trying to catch our babysitter window. I’ve been calling it and nothing. My wife thinks it might be in the car but I already checked. I really need it, I’ve got a work thing this afternoon. Is there any way someone could look?”

Problem Statement 2: “I’m calling about a charge on my account. Actually two charges, I think, from last Tuesday. I’m on the app right now — here, okay so I see one for forty-two eighty on the fourteenth and then there’s another one also from you guys, I think, also on the fourteenth but this one says forty-two eighty-three? I don’t know if that’s a different thing or a mistake. I might have also bought something at the bar, I don’t completely remember, I’m sorry. I just don’t want to get charged for something that wasn’t me. I’m also not sure if that three-cent difference matters, like is that normal?”

Problem Statement 3: “I don’t really know how to say this. We were there last night, my husband and I, for our anniversary actually. And everything was fine — the server was lovely, she was great — but the food. I ordered the duck and it was cold. Like, not a little cold, noticeably cold. And I didn’t say anything because it was our anniversary and I didn’t want the whole thing to be a big deal, but the more I think about it the more I feel like... I don’t know. We saved up for that dinner. I’m not trying to get anything for free. I just wanted someone to know.”

Problem Statement 4: “Yes, I’m looking for whoever handles vendor relationships or operations — I’m not sure exactly what the right department is. I work with companies like yours on the supply chain side, and I’ve had some pretty good results with a few of your competitors, I don’t want to name names, and I’d love to just have a fifteen-minute conversation with the right person to see if there’s a fit. I know everyone’s busy, I just think this could be really useful for your team. Who would be the best person to point me toward?”

Problem Statement 5: “I’m trying to put together an investor deck and I keep getting stuck on the slide where I have to explain what we do. We build — okay, so we work with nonprofits. And we help them with data. But it’s not just data, it’s more like... we help them understand what their data is telling them so they can make better decisions, and we also help them fix the ways they collect data in the first place so the decisions are based on accurate information. So it’s part analytics and part process consulting and

part software, kind of. I know that sounds complicated. Every time I try to make it shorter it sounds like we do nothing, and every time I try to make it complete it sounds like we do everything.”

For each problem statement, write a single solution statement — one or two sentences naming the resolved condition. Then, for each, write a brief note (two or three sentences) explaining what in the problem statement most clearly pointed toward that solution statement, including any detail that would be easy to miss.

Key Takeaways

The central equation of AI workflow design is: problem statement = solution statement. The two sides are equivalent — different forms of the same situation, one unresolved and one resolved.

A messy problem statement is not a failed communication. It is the problem as live human speech expresses it. The messiness contains signal: uncertainty that is real, emotion that reveals stakes, side comments that disclose secondary concerns. Cleaning it up reduces the signal.

The solution statement names the condition that would make the problem no longer a problem. It is not a plan, not a set of steps, not a routing destination. It is a state of affairs — precise and specific enough to evaluate a resolution against.

Method 1 is only about this equation. It ends when the problem statement has been fully heard and the equivalent solution statement has been named. Routing, field design, operational planning, and workflow construction are downstream activities that must wait until the solution statement exists.

Building a workflow without a clearly named solution statement produces a workflow aimed at a vague target. The goal is not fewer words. The goal is fewer wrong workflows. # Chapter 4: Pattern, Authority, and False Completion

Opening Thesis

AI can complete from pattern. AI must verify from authority. These two capacities are not interchangeable, and conflating them is one of the most dangerous failure modes in AI workflow design. The distinction is not subtle. It is not a matter of degree. It is categorical: pattern-based completion and authority-verified completion are different kinds of acts, and designing as if one were the other produces a specific, recurring, and consequential failure — false completion.

This chapter defines both capacities precisely, shows what each looks like in the recurring examples from this book, defines false completion and the conditions under which it occurs, and introduces the two tools that production-safe AI workflow design requires: status language and authority mapping.

Pattern

Pattern is what AI knows from training. It is the accumulated structure of every conversation, document, transaction, process, exchange, and record that was used to build the model. When AI encounters a new situation, it is recognizing and extending patterns from this training — completing the shape of the thing in front of it based on the shapes of similar things it has encountered before.

This capacity is remarkable and, when correctly deployed, immensely useful. It means that AI can recognize what a lost-item report looks like without ever having processed your particular lost-item system. It can recognize what a refund request sounds like without knowing the specific policy of your organization. It can produce a contract draft that is structurally plausible, a product description that is appropriately formatted, a summary of a complaint that captures the key elements. It can do all of this quickly, consistently, and at scale.

Pattern-based completion is AI's most visible and most frequently used capacity. It is what makes AI useful in the emergence phase, where it follows an unfinished sentence to a plausible conclusion. It is what makes AI useful in discovery work, where it can draft the structure of a workflow based on described inputs and outputs. It is what makes AI useful in generation tasks, where it produces language, structure, and form from description.

But pattern-based completion has a hard limit. That limit is authority.

Authority

Authority, in this context, means the externally granted capacity to make a state real. Authority is not knowledge. Authority is not pattern. Authority is not plausibility. Authority is the standing, held by a specific person or system, to enact a specific change in the real world.

When a refund is issued, authority has acted. A person or system with the standing to move money has moved it. The account balance has changed. The transaction exists. The state is real.

When a lost phone is confirmed found, authority has acted. A staff member with direct access to the physical space has examined the found items, matched the description, and confirmed the match. The phone exists in a specific location. The state is real.

When a reservation is confirmed, authority has acted. The reservation system has been updated, the entry exists in the record, the table or room or appointment has been allocated. The state is real.

When a contract is accepted, authority has acted. A person with the legal standing to commit the organization has provided a signature or its legal equivalent. The agreement exists as a binding document. The state is real.

In every case, authority is specific. It belongs to a particular person or system. It applies to a particular kind of act. It produces a change in the real world that would exist even if no one said anything about it. The change does not require a statement to be true — it requires an act.

Pattern cannot substitute for authority. AI can produce the draft of the contract, but the contract is not signed. AI can generate the refund language, but the account has not been credited. AI can write the confirmation message, but the reservation system has not been updated. These are not equivalent states. They differ in one fundamental respect: one of them is real in the world, and the other is a pattern-based representation of what the real version would look like.

The Four Examples: Pattern and Authority

Working through the recurring examples illustrates the distinction precisely and practically.

Lost-Item Report and Found Phone

AI can produce a lost-item report that is structurally complete, clearly formatted, and contains all the relevant fields: the description of the item, the time and date of the visit, the table or area where it was left, the contact information for the guest. AI can draft this report from the guest's unfinished, uncertain, emotionally charged description. This is pattern-based completion, and it is genuinely useful.

But the found phone is not a pattern. The found phone requires a specific act in the physical world: a staff member goes to the lost-and-found, examines the items there, identifies an object matching the description, and confirms the match. That act cannot be completed by pattern. It requires a person with direct access to the physical space — a person with the authority of physical presence — to perform the verification.

The lost-item report is a pattern-based output. The found phone is an authority-verified state. The workflow that moves from one to the other passes through a hard authority requirement, and no AI capability shortens or bypasses that requirement.

Billing Review and Issued Refund

AI can review a billing dispute with sophistication. It can identify the relevant transactions, compare them to the stated description, note anomalies, calculate whether the amounts are consistent with a double charge versus a hold-then-charge sequence, and produce a reasoned assessment of whether an error likely occurred. This is pattern-based completion — an impressive and useful form of it.

But the refund is not a pattern. The refund requires an authorized person or system to move actual money: to credit the account, to initiate a reversal, to process the transaction through the payment system. That act requires authority. The assessment that an error occurred does not constitute the refund. The decision to issue a refund does not constitute the refund. The initiation of the refund process does not constitute the refund. The refund is constituted by the moment the authorized system executes the transaction.

The billing review is pattern-based. The issued refund is authority-verified. They are not equivalent, and a workflow that claims the refund has been issued when the billing review is complete has not resolved the problem — it has generated a false completion.

Contract Draft and Signed Agreement

AI can draft a contract. Given the parameters of an agreement, the relevant legal context, the parties involved, and the key terms, AI can produce contract language that is structurally sound, appropriately formatted, and legally plausible. This is a significant pattern-based capability that has genuine value in legal and business workflows.

But the signed agreement is not a pattern. The signed agreement requires a person with the legal authority to commit their organization — and in many jurisdictions, requires a specific form of signature, a specific process of review and approval, and a specific set of preconditions — to provide their assent in a legally recognized form. No AI capability substitutes for this. The draft is not the agreement. The agreed draft is not the signed agreement. The signed agreement in one party's system is not the countersigned agreement.

Each of these is a distinct state. Pattern produces the draft. Authority creates each subsequent state. The workflow must honor this sequence.

Product Image and Verified Inventory

AI can generate a product image. Given a description, a style, and a set of parameters, AI can produce a visual representation of a product that is accurate to the description provided. In some cases, this representation is so accurate that it closely resembles the actual product.

But verified inventory is not a pattern. Verified inventory requires a specific act: a person with access to the actual stock checks the actual location and confirms that the actual item is present in the actual quantity. A product image, however accurate, is not inventory. An inventory record in a database is not verified inventory until the physical verification has occurred. A system that claims an item is in stock based on pattern completion — based on what the stock level usually is, or what it was when last recorded — is not providing verified inventory. It is providing a pattern-based estimate.

This distinction matters in practice whenever a fulfillment commitment depends on inventory accuracy. The customer who is confirmed that the item is available, based on pattern rather than verification, may arrive to collect it and find the shelf empty. False completion.

False Completion: A Precise Definition

False completion is what happens when AI claims, implies, or emotionally suggests that an authority-bound state exists before authority has made it real.

The definition has three important components.

First: claims, implies, or emotionally suggests. False completion is not only explicit false statement. It includes implication — saying something that logically entails a false completion — and emotional suggestion — using language, tone, or framing that produces in the listener the feeling that an authority-bound state exists, even without explicitly stating it. All three are false completion.

Second: an authority-bound state. False completion is specifically about states that require authority to be real. A pattern-based state — a draft, an assessment, a plausible output — cannot be falsely completed in this sense, because its status as pattern-based is intrinsic to its nature. But an authority-bound state — a refund, a found item, a confirmed reservation, a signed agreement — can be falsely completed, because its existence in the world depends on an act of authority that has not yet occurred.

Third: before authority has made it real. False completion is temporally specific. It is not false to say, after the refund has been processed, that the refund has been issued. It is false to say so before the refund has been processed, regardless of how certain the eventual processing may be.

This precision matters because many instances of false completion are not intended as deception. They are intended as reassurance. The AI generates warm language because it is completing a conversational pattern in which the customer expects good news after a complaint. The pattern says: acknowledge the complaint, apologize, confirm the resolution. The AI generates the confirmation because that is the natural completion of the pattern. The problem is that the confirmation describes a state that does not yet exist, because the authority act has not yet occurred.

The Four False Completion Examples

“Your Refund Has Been Issued”

A customer reports a billing dispute. The AI reviews the account, determines that an error likely occurred, and says: “I can see what happened here — there was indeed a duplicate charge. Your refund has been issued and should appear in your account within three to five business days.”

If the refund has been initiated by an authorized system, this statement is accurate. If the billing review is complete but no authorized system has yet executed the transaction, this statement is false completion. The customer now believes their account will be credited. They may spend against that expectation. When the credit does not appear because the authority act did not actually occur — or occurred later than implied — the customer’s situation is worse than before the false completion.

“We Found Your Phone”

A customer calls to report a lost phone. The AI processes the report, matches the description against a database of found items, identifies a plausible match, and says: “Good news — it looks like we found your phone. We’ll hold it at the front desk for you.”

If a staff member has physically identified and verified the match, this statement is accurate. If the match is based on a database entry, a description match, or a pattern-based inference — without physical verification by a person with direct access to the lost-and-found — this statement is false completion. The customer drives to the restaurant. The item at the front desk is not their phone. Or the match was to an item belonging to a different guest. Or the item was not actually there when the staff member checked. The customer’s urgency (they had a work call) has now been compounded by a wasted trip.

“You Are Confirmed”

A customer books a reservation through an AI-assisted booking workflow. The AI processes the request, confirms the availability, and says: “You’re all set — you’re confirmed for Saturday at seven-thirty.”

If the reservation system has been updated with a real record, this statement is accurate. If the AI has processed the request but the update to the reservation system failed, lagged, or was queued but not executed — and the AI generated the confirmation from the pattern of what a confirmation message looks like — this statement is false completion. The customer arrives Saturday at seven-thirty. Their name is not in the system. There is no table.

“The Agreement Is Accepted”

A vendor submits a proposal. An AI-assisted review workflow processes the proposal, compares it to criteria, generates a positive assessment, and says: “Your proposal has been reviewed and we’re pleased to accept. You should receive a formal agreement shortly.”

If an authorized decision maker has reviewed and accepted the proposal, this statement is accurate. If the AI has completed the pattern of what a favorable review and acceptance looks like — without authorization from a person with the standing to commit the organization — this statement is false completion. The vendor begins work. The organization’s authorized decision maker eventually reviews the proposal and declines it. The vendor has expended resources based on a false completion.

Status Language as the Production-Safe Solution

The production-safe solution to false completion is status language — a precise vocabulary for describing exactly where in the authority chain a situation currently exists.

Status language does not replace authority. It describes the current position relative to authority, accurately and at every step. It makes the workflow’s position in the authority chain legible to the human, so they know what has been done and what has not yet been done, and can calibrate their own expectations accordingly.

A working status vocabulary for most service workflows moves through these stages:

Drafted — a document, report, request, or record has been produced but has not been submitted or acted upon. The output exists as a pattern-based completion. No authority has acted.

Prepared — the output has been reviewed, verified, and is ready for submission or next action. It is accurate and complete as far as can be determined without authority acting.

Submitted — the request, document, or report has been formally transmitted to the relevant person or system. It has entered the authority chain but has not yet been acted upon.

Routed — the submission has been directed to the specific person or system with the relevant authority. It is no longer in a general queue.

Pending verification — the relevant authority is in the process of verifying the claim, description, or request against the actual state of affairs. This is the step that pattern cannot perform: actual physical or system verification.

Verified — the authority has confirmed that the described state matches reality. The phone matching the description has been physically examined. The transaction has been checked against the actual account. The inventory has been physically confirmed.

Approved — the authority with decision-making standing has approved the action. The refund has been approved. The agreement has been approved.

Issued — the approved action has been executed by the relevant system. The refund has been processed. The credit is real.

Confirmed — the state has been communicated to all relevant parties and acknowledged as real. The customer knows the refund is in their account.

Completed — the full arc from problem statement to resolution is finished. The situation is resolved.

These stages are not bureaucratic formalism. They are precision instruments. Each one names a different real state of affairs. Using them correctly means that the human at every step of the workflow knows exactly where their situation stands — what has happened and what has not yet happened — without being given the feeling of a resolution that does not yet exist.

Status is not bureaucracy. Status is truthfulness.

Authority Maps

A workflow that uses status language correctly requires that the designer knows, before building the workflow, where every authority-required step is located. This knowledge is produced through authority mapping.

An authority map is a complete identification of every step in the workflow that requires an act of authority to change the state. For each such step, the map identifies: what kind of authority is required (physical verification, system access, decision-making standing), who or what holds that authority, and what the preconditions are for the authority to act.

Authority mapping is not glamorous. It is not the part of AI workflow design that produces impressive demonstrations or visually compelling outputs. It is, however, the part that determines whether the workflow is truthful — whether every status statement the AI makes corresponds to a real state in the world.

The designer who skips authority mapping builds a workflow with implicit assumptions about where authority lives. Those assumptions may be correct for the typical case. They will be wrong for edge cases, exception paths, system failures, and the many situations in which the expected authority holder is unavailable, the system is down, the physical verification cannot be completed in the expected time, or the decision maker has a different view than the pattern would predict.

Authority mapping converts those implicit assumptions into explicit design decisions — and explicit design decisions can be tested, validated, corrected, and maintained in a way that implicit assumptions cannot.

The Distinction Between Discovery and Production

This chapter has introduced the distinction between pattern and authority in the context of production workflows — real AI systems deployed to serve real humans with real problems. In this context, false completion is a failure mode that damages trust, creates operational problems, and harms the humans the system is supposed to serve.

But this is not the only context in which AI workflow design operates. In Method 2 — Hallucination, the second method in the life cycle — the controlled use of pattern-based completion for discovery purposes is not a failure. It is a design tool. Letting AI complete a pattern toward an ideal outcome, in a context where everyone knows that the completion is exploratory rather than authoritative, reveals possibilities that structured analysis might miss.

The distinction is one of context and intent. In discovery, pattern-based completion is used deliberately, transparently, and for a specific purpose: to reveal the shape of a possible future state. The completion is clearly marked as completion. No one is told that their refund has been issued when it has not. No authority-bound claim is made.

In production, pattern-based completion must be bounded by authority verification. The completion can do the drafting, the structuring, the language generation. But any claim about a state that requires authority to be real must wait for the authority act.

This distinction will be developed fully in Part Two, where Method 2 is addressed directly. It is introduced here to prevent a misreading of this chapter’s argument: the point is not that pattern-based completion is bad. The point is that it is bounded — and that the boundary must be designed in explicitly, because AI’s conversational capacity will otherwise complete the pattern past the boundary without noticing that a boundary exists.

Common Designer Mistake

Enthusiastic Language That Emotionally Confirms a False State

The most common form of false completion in production AI systems is not overt misstatement. It is warm, well-intentioned language that produces in the human the emotional experience of resolution before resolution has occurred.

The pattern is recognizable. A customer reports a problem. The AI listens attentively. The AI acknowledges the concern. The AI says something like: “Absolutely, I completely understand — we’re going to take care of this for you. You won’t be charged for that, I promise.” The human feels heard. The human feels resolved. The human hangs up with confidence.

The account has not been credited. No authority has acted. The promise was generated by pattern — completing the conversational arc in which acknowledgment is followed by assurance is followed by resolution. The pattern is accurate as a description of how the conversation should go. It is false as a description of what has happened.

The designer who built this workflow may have intended exactly this outcome — warm, reassuring language that leaves the customer feeling good. The UX goal was achieved. The operational goal was not. The customer will return when the credit does not appear, with a grievance that is now larger because it includes the false assurance they were given in addition to the original problem.

This mistake is disguised as good UX because the metric most commonly used to evaluate AI interactions — customer satisfaction immediately following the conversation — will show a positive result. The human was satisfied when they hung up. The satisfaction was based on false completion. The downstream consequences will appear later, when they appear in a different metric.

The correction is to build status language into the warmth. The AI can still be warm. It can still be reassuring. But the reassurance must describe a real state: “I’ve submitted this to our billing team for review — you should hear back within 24 hours with the result. Here’s your case reference number.” Warm, specific, and accurate.

Design Practice

Status Language for the Lost-Phone Workflow

Take the lost-phone scenario and work through five distinct moments in the workflow. For each moment, write the AI’s response using accurate status language. Your response should be warm and clear — it should not feel clinical or bureaucratic — but it must not claim an authority-bound state that does not yet exist.

Moment A: The Report Is Filed The guest has just finished describing the lost phone. The AI has gathered all relevant information: the description, the approximate location, the time of the visit, the contact number. What does the AI say at this moment?

Write a response that accurately describes what has been accomplished (the report exists, has been logged, has been submitted to the staff who handle lost items) without claiming that the phone has been found, identified, or will definitely be recovered.

Moment B: A Found Item Is Reported by a Staff Member A staff member has found a phone matching the general description and reported it to the system. The match is plausible but not yet physically verified against the specific description the guest provided. What does the AI say to the guest when it reaches out about this?

Write a response that accurately describes the state (a possible match has been identified) without claiming verification (without saying “we found your phone”).

Moment C: The Item Is Verified as Matching A staff member has physically compared the found

phone to the guest's description and has confirmed that it is a match: same case, same make, screen-locked, rings when called from the partner's number. What does the AI say now?

Write a response that accurately describes the verified state and what the guest needs to do next.

Moment D: The Guest Has Been Called to Retrieve the Item The guest has been contacted and has confirmed they are coming to retrieve the phone. They are on their way. What does the AI say?

Write a status confirmation that accurately describes what is true: the match is verified, retrieval is scheduled, the item is being held.

Moment E: The Item Is in the Guest's Hands The guest has arrived, shown identification, and taken possession of the phone. The staff member has confirmed the transaction. What does the AI say to close the loop?

Write a completion statement that is warm, accurate, and marks the full arc from problem statement to resolution.

After writing all five responses, review them as a sequence. Does each response describe exactly what is true at that moment — no more and no less? Is there any moment at which a reasonable reader might take the language to imply an authority-bound state that has not yet occurred? Revise any such moment.

Key Takeaways

AI can complete from pattern. Pattern is what AI knows from training — the accumulated structure of similar situations that allows AI to recognize, extend, and generate plausible outputs. Pattern-based completion is AI's most visible capacity and the source of much of its value.

AI must verify from authority. Authority is the externally granted capacity to make a state real — the standing held by a specific person or system to enact a specific change in the world. Authority produces states that exist in reality regardless of what is said about them.

Pattern and authority are not interchangeable. A pattern-based completion of an authority-bound state — a refund draft, a found-item description, a confirmation message — is not the authority-bound state. They are categorically different things.

False completion happens when AI claims, implies, or emotionally suggests that an authority-bound state exists before authority has made it real. False completion damages trust, creates operational failures, and harms the humans the system is meant to serve. It is a design failure, not a tone problem.

Status language is the production-safe solution. A precise status vocabulary — Drafted, Prepared, Submitted, Routed, Pending verification, Verified, Approved, Issued, Confirmed, Completed — allows every statement about a situation's state to be accurate at every moment. Status is not bureaucracy. Status is truthfulness.

Authority maps identify every step in a workflow that requires an act of authority to change the state. They convert implicit assumptions about authority into explicit design decisions that can be tested, vali-

dated, and maintained.

In discovery contexts, controlled pattern-based completion is a design tool. In production contexts, it must be bounded by authority verification. The boundary must be designed in explicitly, because AI will complete the pattern past the boundary without noticing that it exists. # Part Two: The Life Cycle

There is a version of AI workflow design that treats each engagement as a collection of decisions — what to prompt, what to route, what to automate, what to escalate. That version produces workflows that solve the wrong problems efficiently.

Part Two offers a different structure. It is built around five methods that together form a life cycle — not a project plan, not a phase chart, not a handoff sequence. A life cycle. That word is chosen deliberately. It implies recursion. It implies renewal. It implies that what ends is not finished, but feeding forward into what comes next.

The five methods are:

Method 1: Emergence gives you the solution. Method 2: Hallucination gives you the ideal. Method 3: As-is discovery gives you reality. Method 4: To-be selling gives you transformation. Method 5: Stewardship gives you renewal.

Each method has one primary output and one primary purpose. Each method depends on the output of the method before it. This is not arbitrary sequencing — it is load-bearing. Emergence must come before hallucination because you cannot simulate the ideal workflow without knowing what you are solving. Hallucination must come before as-is discovery because you need the to-be model in hand before you start interviewing the client — otherwise the client’s reality collapses your vision of the ideal before you ever generate it. As-is discovery must come before to-be selling because you cannot present a gap until you have both sides of it. And stewardship must come after deployment not as an afterthought, but as a recurring design discipline that asks whether the workflow still earns its existence.

Chapter 5 gives you the full view — all five methods in relationship to each other, the failure modes that appear when any method is skipped, and the recursive structure that distinguishes this life cycle from a waterfall.

Chapters 6 through 10 give you one method each, in full depth: the purpose, the mechanics, the risks, the designer’s role, and a concrete worked example.

By the time you finish Part Two, you should be able to look at any AI workflow engagement — real or hypothetical — and locate it on the life cycle. You should be able to name which methods were used well, which were skipped, and what the skipping cost. You should be able to run all five methods yourself, in order, with discipline.

The life cycle is the spine of this book. Everything before Part Two prepared you to understand why each method exists. Everything after Part Two will ask you to apply the life cycle under harder conditions. But it starts here, with a structure that is rigorous enough to lean on.

Chapter 5: Overview of the Life Cycle

Opening Thesis

Every AI workflow design engagement moves through five methods. The order matters. Skipping any method produces predictable failure modes — not random ones, not unlucky ones, but the specific kind of failure that corresponds to the specific method that was skipped. This chapter maps the five methods as a system: what each one does, why each one depends on the one before it, and what happens to the workflow — and to the client — when the sequence is violated.

Understanding the life cycle as a system is more important than mastering any single method in isolation. A designer who is excellent at method 2 but skips method 1 will generate a brilliant workflow for the wrong problem. A designer who is thorough in methods 1 through 4 but treats method 5 as an afterthought will deliver a workflow that degrades silently for months before anyone notices. The methods are individually necessary; together they are sufficient.

The Five Methods

Method 1: Emergence Gives You the Solution

Before any workflow can be designed, the problem must fully surface. This sounds obvious, and it is routinely ignored. Emergence is the practice of allowing the human to fully express the problem before the AI — or the designer — attempts to classify, route, resolve, or respond to it. The output of method 1 is exactly one thing: the problem statement and its equivalent solution statement. The problem statement names the full, messy, human-language description of what is wrong. The solution statement names the condition that would make the problem no longer a problem.

Emergence gives you the solution. Not by analyzing the problem. Not by diagnosing it. By allowing the full shape of it to appear before any design work begins.

The lost phone is a useful example throughout this book. The problem statement is not “a customer called to report a lost phone.” It is the full expression: “I was at your restaurant last night, I’m pretty sure I left my phone on the table, I’ve already tried calling it and no one answered, I’m panicking because it has everything on it, I’m supposed to travel tomorrow.” The solution statement is: the phone is in the

customer's possession and they know it is secure. Every word between those two states is where the workflow lives. But you cannot design that workflow until you have both endpoints clearly named.

Method 2: Hallucination Gives You the Ideal

Once you have the solution statement, you do not go to the client. You go to the AI first. Method 2 is a controlled simulation: the AI is given the solution statement and asked to generate the best-class workflow for reaching that solution — without knowing anything about the client's current reality. This is called a blind ideal workflow simulation. The term “hallucination” refers to the AI's characteristic of generating fluent, confident, plausible output without being grounded in the client's specific constraints.

In production, hallucination is false completion. In discovery, hallucination reveals the ideal. Method 2 is the moment in the life cycle when this capacity is not a hazard — it is the engine. The designer controls it by giving the AI a clear solution statement and explicitly withholding all information about the client's current operations.

The output of method 2 is a to-be model: a best-class representation of how the workflow would run if all the necessary systems, authority paths, records, notifications, and staff behaviors existed. This model becomes the interview guide for method 3.

Method 3: As-Is Discovery Gives You Reality

Only after the to-be model is complete does the designer engage the client to understand the current state. The to-be model, generated in method 2, functions as the interview scaffold. Each step in the to-be model generates a discovery question: Do you have this system? Who owns this step? What actually happens here? What breaks during rush hours?

As-is discovery gives you reality. Not a sanitized version of reality. Not the official process as written in the employee handbook. The real process: the informal workarounds, the authority gaps, the staff member everyone calls because she remembers everything, the shelf in the back where lost items accumulate with no record attached to them.

The output of method 3 is a documented as-is model annotated with gap markers — every place where the current reality falls short of the to-be model. These gaps are not failures. They are the material the designer works with in method 4.

Method 4: To-Be Selling Gives You Transformation

Method 4 is where the designer presents the gap between the to-be model and the as-is model — and sells the client on a specific, achievable path toward the to-be state. This is not the same as selling AI capabilities or demonstrating a voice agent. To-be selling gives you transformation. The transformation is the movement from as-is toward to-be, and the designer is responsible for both defining that movement and helping the client commit to the process changes required to enable it.

A commodity AI provider automates the current process. A serious AI workflow designer sells movement from as-is toward to-be. This distinction is the commercial and ethical difference between a vendor

and a designer. The vendor asks: what do you do now, and how can I make a computer do it? The designer asks: here is what best-class looks like, here is where you are, here is how far you can realistically move, and here is what has to change — in your staff behavior, your record systems, your authority structure — to get there.

The output of method 4 is an agreed transformation target: the workflow the client is willing and able to support, which is the best reachable position toward the to-be model given current resources, capacity, and organizational will.

Method 5: Stewardship Gives You Renewal

Deployment is not the end of the life cycle. It is the beginning of method 5. Stewardship is not maintenance. Maintenance preserves the workflow. Stewardship renews the workflow around the solution statement. The question maintenance asks is: is it still running? The question stewardship asks is: does it still best serve the solution statement?

Stewardship gives you renewal. Renewal does not mean constant change. It means a disciplined, recurring commitment to asking whether the current workflow is still the best path from problem statement to solution statement — given what has changed in the AI models, in the client’s business, in customer behavior, in available tools, in cost and latency profiles. If the answer is yes, the current workflow is confirmed, not by inertia but by examination. If the answer is no, the workflow is renewed, not because it broke, but because something better is now possible.

Why the Order Matters

The five methods are not interchangeable. Each method depends on a specific input that only the previous method can produce.

Method 1 must precede method 2 because the to-be simulation requires a solution statement as its target. If you run a simulation without a solution statement, you generate a workflow pattern that may be plausible but is aimed at nothing specific. The simulation will look thorough and may even be reused in the next engagement — which is exactly how you end up deploying workflows that solve a different client’s problem.

Method 2 must precede method 3 because the to-be model functions as the interview scaffold for discovery. If you conduct the discovery interview before generating the to-be model, two problems occur. First, you will ask incomplete discovery questions — because you will not yet know what gaps you are looking for. Second, the client’s description of their current reality will begin to contaminate your vision of the ideal. You will hear “we just write it on a sticky note” and begin designing around sticky notes. The to-be model becomes a slightly improved version of the as-is reality rather than a representation of best-class outcomes. Do not contaminate the to-be model too early.

Method 3 must precede method 4 because you cannot present a gap without both sides of it. To-be selling

requires showing the client exactly where their current process diverges from the ideal. Without the as-is map, the designer is selling a vision with no concrete anchor in the client's reality. The client will agree enthusiastically and then encounter the actual change required when implementation begins — which is where most engagements quietly collapse.

Method 4 must precede deployment because the agreed transformation target is the design specification. Deploying without that agreement means deploying against the designer's best guess about what the client wants and is capable of supporting. That guess is almost always wrong in at least one significant dimension.

Method 5 must follow deployment because the workflow will be tested against conditions that method 4 never fully anticipated. Stewardship begins the moment the workflow is in production and does not end until the workflow is retired.

The Recursive Nature of the Life Cycle

The life cycle is not a waterfall. Waterfalls end. The life cycle renews.

Method 5 feeds back into method 1. After a stewardship review, the designer may determine that the current solution statement still holds — in which case the renewal is focused on methods 2 through 4, generating a new to-be model and a new transformation path. Or the stewardship review may reveal that the problem itself has evolved: the client's business has changed, or customer behavior has shifted, or the original solution statement no longer fully describes what resolution means. In that case, the cycle begins again at method 1. A new emergence. A new solution statement. A new to-be model.

This recursive structure is what distinguishes AI workflow design from software development as traditionally practiced. Software projects have releases. AI workflow engagements have ongoing cycles. The discipline is not to complete the life cycle once and move on. The discipline is to run the life cycle with enough rigor that each return to method 1 is informed by everything learned in the previous pass through methods 2 through 5.

Two Modes of Life Cycle Use

The five-method life cycle operates in two distinct modes, both of which are legitimate and both of which require the same structural discipline.

The full engagement mode involves all five methods applied to a new client or a substantially new problem. The designer runs emergence, generates the to-be model, conducts discovery, presents the transformation, and establishes stewardship cadence. This is the appropriate mode for new engagements, major redesigns, and any situation where the solution statement is not yet established.

The live production loop involves methods 1 and 5 operating continuously while the workflow is in production. Method 1 runs in every customer interaction: emergence happens live, the solution statement is identified in real time, and the workflow attempts to move toward resolution. Method 5 runs in the background: the steward monitors, reviews, identifies incidents, and schedules renewal. Methods 2, 3, and 4 are held in reserve — activated when stewardship determines that renewal requires generating a new to-be model and a new transformation path.

Understanding which mode you are in at any given moment is part of the designer’s ongoing discipline.

Failure Modes by Skipped Method

Each method, when skipped, produces a specific and predictable failure. These are not hypothetical — they are documented patterns in AI workflow deployments that have reached the designer’s attention through post-mortems, client complaints, and renewal conversations.

Skipping method 1 (no emergence): The designer starts with method 2, generating a to-be model based on an assumption about what is being solved. The workflow is technically excellent and completely misaligned with the actual problem. Customers complete the workflow and feel unheard. The solution statement was never named, so no one can articulate why the workflow is failing.

Skipping method 2 (no ideal simulation): The designer moves from emergence directly to discovery, generating the as-is map without a to-be model in hand. The discovery interview reveals current reality, which becomes the design specification. The resulting workflow is a well-documented automation of the current broken process. The client wonders why they paid for this.

Skipping method 3 (no discovery): The designer generates the to-be model and moves to selling and deployment without understanding the current reality. The workflow assumes systems that do not exist, authority structures that have not been defined, and staff behaviors that have not been established. Deployment fails immediately or delivers inconsistent outcomes because the underlying business process cannot support the workflow’s requirements.

Skipping method 4 (no to-be selling): The designer generates the to-be model and the as-is map, then deploys the workflow without presenting the gap or securing commitment to the required process changes. The workflow is technically correct but the client’s operations do not change to support it. Staff continue informal workarounds. Records are not created. Authority is not exercised. The workflow works perfectly in controlled tests and fails in the real world.

Skipping method 5 (no stewardship): The workflow is deployed and the engagement is closed. Over time, the AI models change. The client’s business evolves. New capabilities become available. The workflow drifts from alignment with the solution statement. Eventually a customer encounter makes the misalignment visible — but by then the drift is significant and correction is expensive.

Common Designer Mistake

Treating the Life Cycle as a Waterfall

The most common structural error made by designers encountering this framework for the first time is treating the five methods as a linear sequence to be completed once. In this misreading, method 1 is a kickoff meeting, method 2 is a design sprint, method 3 is a discovery phase, method 4 is a sales presentation, and method 5 is a support contract. The engagement closes when method 5 begins.

This is not the life cycle. It is the life cycle flattened into a project plan.

The error matters because it produces a specific kind of organizational behavior: the designer treats stewardship as a post-sale obligation rather than a design discipline. Stewardship reviews are reactive — triggered by client complaints rather than by the steward's proactive renewal questions. When the cycle restarts, it restarts as a new project rather than as a continuation of the previous cycle. The knowledge accumulated in methods 3 and 4 — the as-is map, the gap analysis, the transformation agreement — is lost rather than carried forward.

The life cycle is recursive. Method 5 feeds back into method 1. The discipline is ongoing. The goal is not fewer words. The goal is fewer wrong workflows.

Design Practice

Map an Engagement Against the Five Methods

Select a recent or hypothetical AI workflow engagement. It may be an engagement you participated in, observed, or can reconstruct from available information. It may be a hypothetical built around one of the recurring examples in this book — the lost phone, the double charge, the bad restaurant experience.

For that engagement, work through the following questions in writing.

First: where did the engagement begin? At which method did the designer (or the organization) actually start? Was there an emergence phase, or did the engagement begin with a statement of the problem as the client understood it? Was there a to-be simulation, or did the design proceed directly from the stated problem to a proposed solution?

Second: which methods were skipped or compressed? Were they skipped intentionally, with awareness of the tradeoff? Or were they skipped because the designer did not know they were missing?

Third: what was the result? Identify the specific failure mode that corresponds to each skipped method. If all methods were present, identify any places where the sequence was violated — where method 3 contaminated method 2, for example, or where method 5 was treated as a support function rather than a renewal discipline.

Fourth: if you were to run this engagement again from the beginning with full awareness of the five-method life cycle, where would you start? What would you do differently in each method?

This exercise is not retrospective criticism. It is pattern recognition. The goal is to internalize the life cycle well enough that you can locate any engagement on it at a glance — and know, immediately, which method you are in and which one comes next.

Key Takeaways

The five-method life cycle — emergence, hallucination, as-is discovery, to-be selling, and stewardship — is the structural backbone of AI workflow design. Each method produces one primary output that the next method requires as input. The sequence is not arbitrary; it is load-bearing.

Skipping any method produces a specific and predictable failure mode. Skipping emergence produces a workflow aimed at the wrong problem. Skipping hallucination produces an automated version of the broken as-is process. Skipping discovery produces a workflow that assumes systems and behaviors that do not exist. Skipping to-be selling produces a workflow that the client's organization cannot support. Skipping stewardship produces a workflow that drifts silently from alignment with the solution statement.

The life cycle is recursive, not linear. Method 5 feeds back into method 1. The discipline is ongoing. A designer who treats the life cycle as a waterfall will complete the sequence once and deliver work that degrades over time without a structural mechanism for renewal.

The two modes of life cycle use — full engagement and live production loop — require the same five-method discipline applied at different scales and frequencies. Knowing which mode you are in is part of the designer's ongoing awareness.

Early patience prevents late repair. Running all five methods rigorously at the beginning of an engagement is not overhead. It is the difference between a workflow that holds and one that breaks in the field.

Chapter 6: Method 1 — Emergence Gives You the Solution

Opening Thesis

Method 1 has one output and one output only: the problem statement and its equivalent solution statement. Nothing else is built in Method 1. No workflow is designed. No tool is selected. No routing logic is sketched. No database schema is contemplated. No form fields are imagined. The entire purpose of Method 1 is to allow the problem to fully surface through live human speech — and then, and only then, to name the solution statement that becomes the design target for everything that follows.

This sounds like a modest ambition. It is not. The failure to conduct a genuine emergence phase is the most common structural error in AI workflow design, and it is responsible for a larger proportion of failed deployments than any other single mistake. The error is not usually carelessness. It is eagerness — the designer’s natural impulse to begin solving before the problem has finished arriving.

The beginning of the workflow is not extraction. The beginning of the workflow is emergence.

What Emergence Means

Emergence, in this context, means allowing the full problem statement to come forward through live human speech before any attempt is made to classify, extract, route, or resolve. The word “full” is important. A partial problem statement produces a partial solution statement, which produces a workflow aimed at a fragment of what the person actually needs.

Human beings do not arrive with their problems fully formed and neatly packaged. They arrive with a felt sense of something wrong, an urgency, a set of facts they have assembled in whatever order they could manage, and a concern about what might happen if the problem is not resolved. The full problem statement emerges through the act of expressing it — often through repetition, through tangents, through emotional elaboration, through the gradual clarification that comes when someone is genuinely listened to.

Every speaking human arrives unfinished. This is not a deficiency in the human. It is the nature of how

problems are carried and how they surface. A well-designed emergence phase honors this. It gives the human enough space and enough responsive presence to complete the emergence before any action is taken.

The AI's role in emergence is specific: it converses, it holds ambiguity, it reflects briefly, and it allows the human to continue. It does not classify. It does not route. It does not ask the human to select from a menu of problem types. It does not interrupt to say "I understand — it sounds like you need our lost item department." That sentence, however friendly, is a collapse. It replaces the human's full problem with a category name, and the category name is almost always smaller than the problem.

Do not collapse the human too early.

What Method 1 Is Not

Because Method 1 is the first method in the life cycle, and because it involves the AI speaking with a human, designers often confuse it with several adjacent practices that are fundamentally different.

Method 1 is not intake design. Intake design — the construction of structured forms, question sequences, or classification systems meant to capture information at the beginning of an interaction — is built on the assumption that the designer already knows what kind of information needs to be gathered. Intake design presupposes the problem type. Emergence discovers it. The moment you add a structured intake form to Method 1, you have destroyed the emergence phase, because the form imposes categories before the human has finished arriving.

Method 1 is not routing design. Routing design asks: given that the problem is of type X, where should it go? Routing presupposes classification. Classification presupposes a complete problem statement. You cannot route a problem that has not yet fully emerged.

Method 1 is not tool design, API design, or database design. None of these belong in Method 1, not even in preliminary sketch form. The moment a designer begins thinking about what tools will be called, which APIs will be invoked, or what fields will be written to a record, they have left Method 1 — whether or not the emergence phase is complete. These thoughts contaminate the design target. They begin to shape what "solution" means before the solution statement has been properly named.

Method 1 is not classification. Even if the designer can already see, from the first few words of the customer's expression, that this is a lost-item case or a billing dispute, that recognition must be held lightly and kept out of the interaction until emergence is complete. The classification may be correct. But even a correct early classification narrows what the human is allowed to express, and narrowing too early produces an incomplete problem statement.

Absolutely none of the following belong in Method 1: artifacts, records, APIs, tools, forms, databases, alerts, escalation paths, department assignments, or case types. Method 1 has one output. Only one output. The problem statement and its equivalent solution statement.

The AI's Natural Advantage in Emergence

AI, as a conversational participant, has properties that make it well-suited to the emergence phase — properties that human agents, under operational pressure, often cannot sustain.

AI can hold ambiguity without discomfort. A human agent who hears “I think I left my phone, but I’m not sure, it might have been somewhere else” will often redirect toward certainty — “So you think you left it at our restaurant?” — because ambiguity is uncomfortable and efficiency demands resolution. AI, properly designed, can remain in the ambiguous space with the customer, reflecting what it has heard without forcing resolution, allowing more of the problem to surface.

AI can reflect without collapsing. Reflection — repeating back what you have heard in a way that invites continuation — is a specific skill. Done well, it signals to the human that they have been heard and creates the condition for more of the problem to emerge. Done poorly, it becomes a paraphrase that reduces the human’s expression to a summary and signals that the expression phase is over. AI can be prompted to reflect in the first mode: “It sounds like you’re dealing with more than one concern here. Take your time — what else is going on?”

AI can wait. Under heavy call volume, human agents cannot wait. They have a queue behind them. AI does not experience queue pressure in the same way, and its patience does not degrade over the course of a shift. A well-designed AI emergence phase can be as patient at interaction 10,000 as it is at interaction 1.

The designer’s job is to preserve these capacities, not engineer them away. An AI that has been over-engineered with classification prompts, routing logic, and structured intake sequences at the beginning of the interaction has had its emergence capacity replaced with an intake funnel. The designer has done this. The customer pays the cost.

The Golf Analogy: Do Not Help the Ball

There is a useful analogy from golf instruction. Beginning golfers often try to help the ball into the air by lifting the club at impact. The impulse is entirely natural — you can see the ball sitting on the ground, and you want it to go up, so you help it up. But this is precisely the wrong move. The club is designed to send the ball up. If the golfer trusts the club and swings through, the ball rises. If the golfer helps the ball, the swing breaks down and the shot is worse.

The equivalent error in Method 1 is what might be called “helping the AI.” A designer watching the emergence phase unfold may see the problem taking shape before the customer has finished expressing it, and feel the urge to prompt the AI to classify, to route, to respond. The AI, like the golf club, is designed to handle the emergence phase if it is properly configured and left alone. Prompting it to classify or route

mid-emergence is like lifting the club at impact. The motion breaks down. The emergence is incomplete. The problem statement is partial.

Do not help the AI during emergence. If you have configured the emergence phase correctly — a clear instruction to hold ambiguity, reflect without collapsing, and wait for the human to complete — then trust it. The emergence will produce the problem statement. Then you have something to work with.

The Five Examples and Their Solution Statements

The recurring examples in this book are not arbitrary. They are chosen because they represent a range of problem types — logistical, financial, experiential, relational, conceptual — that an AI workflow designer is likely to encounter across many different client engagements. Understanding how emergence works in each case is part of the designer’s pattern literacy.

Lost phone at a restaurant. The problem statement, fully emerged, includes: the time and table of the visit, the phone’s description, the emotional urgency of the customer, the fact that the phone has already been called without answer, the customer’s travel plans that create time pressure, and possibly concerns about personal data. The solution statement is specific: the phone is in the customer’s possession and they know it is secure. The solution statement is not “the customer was helped” or “a lost item report was filed.” It is the condition that makes the problem no longer a problem.

Possible double charge on a credit card. The problem statement, fully emerged, includes: the customer’s uncertainty (they are not certain it is a double charge — they think it might be), the specific transactions they have noticed, their concern about whether this represents a systemic problem with their account, and their underlying anxiety about financial security. The solution statement is: financial security restored. The customer knows what the charges are, has confidence that their account is accurate, and knows that any error has been corrected.

Bad restaurant experience. The problem statement, fully emerged, includes: the specific nature of the bad experience (service failure, food quality, ambiance, a specific interaction with a staff member), the customer’s emotional state, their history with the restaurant (first visit? regular?), what they had hoped the evening would be, and what would make it right in their view. The solution statement is: the experience is made right. Not “the customer was apologized to.” Not “a discount was offered.” The experience is made right — which means understanding what “right” means for this specific customer, which only emergence can reveal.

Confusion about next steps. The problem statement, fully emerged, includes: what the customer is trying to accomplish, what they have already tried, where they are stuck, and what they are afraid will happen if they cannot proceed. The solution statement is: the next step is clear and the customer can take it confidently. This example is valuable because it illustrates that some solution statements are process-level rather than outcome-level — the solution is not a delivered result but a cleared path.

Unclear business idea. The problem statement, fully emerged, includes: the founder’s attempt to de-

scribe what they are building, the points at which the description breaks down or circles back, the distinction between what the founder knows and what they have not yet articulated, and the context in which clarity matters (a pitch meeting, a conversation with a potential hire, a document that needs to go to investors). The solution statement is: the idea is made communicable. Not “the pitch deck is built.” Not “the elevator pitch is refined.” The idea is communicable — which means the founder can explain it to any relevant audience and the listener understands what is being offered.

How to Identify the Solution Statement

The solution statement names the condition that would make the problem no longer a problem. This definition is precise and it is productive. It avoids two common errors.

The first error is naming a solution that is too mechanism-specific. “The phone is returned to the customer via a callback to the number they provided” is a mechanism. “The phone is in the customer’s possession and they know it is secure” is a solution statement. The mechanism is one way to reach the solution statement. There may be others. The workflow designer’s job is to design toward the solution statement, not to prescribe the mechanism prematurely.

The second error is naming a solution that is too abstract. “The customer is satisfied” is not a solution statement. It is a sentiment. Solution statements name conditions, not feelings. “The customer knows what the charges are, has confidence that their account is accurate, and knows that any error has been corrected” is a condition. It can be evaluated. It can be designed toward. It is specific enough to generate to-be steps in Method 2.

The equation is: problem statement = solution statement. These two states are equivalent. The problem statement is the messy side — the full human expression of what is wrong. The solution statement is the resolved side — the condition that makes the problem no longer a problem. They are not opposites. They are the two faces of the same situation. The messy side is not wrong. The resolved side is the design target.

When Method 1 ends, you have named the solution statement. That is all. Not a workflow. Not a classification. Not a routing rule. Just the target. Everything else comes later.

Full Example: A Live Emergence Transcript

The following is a realistic transcript of an AI handling the lost-phone case. Note the AI’s behavior throughout: it reflects without collapsing, holds ambiguity, and does not classify or route until the emergence is complete. Following the transcript, the solution statement is identified. Then a contrast case shows how a poorly designed system would have handled the same opening.

Customer: Hi, I was at your restaurant last night and I think I left my phone there.

AI: I'm sorry to hear that — that's a stressful situation. Tell me more about what happened. What time were you there, roughly?

Customer: It was around 8 o'clock. We were at a table near the back, by the window. I noticed this morning when I woke up that I didn't have it.

AI: Got it. You noticed this morning. And have you been able to reach anyone at the restaurant yet, or is this your first contact?

Customer: This is my first call. I actually tried calling my own phone a few times, but no one picked up. I'm hoping someone found it and turned it in.

AI: That makes sense. I can hear how worried you are. Is there anything else going on with this that I should know about — anything that makes this especially urgent for you today?

Customer: Yeah, I'm flying out tomorrow morning. So if I don't find it today I'm going to be in real trouble. And also — I know this sounds paranoid — but it has a lot of personal stuff on it, so I just want to know it's secure.

AI: That doesn't sound paranoid at all. So you need to know it's found and secure, and you need to be able to get it before tomorrow morning. Those are both completely reasonable. Let me make sure I have the picture right: you were at a table in the back near the window around 8pm last night, you've already tried calling the phone without answer, and you need a resolution today because you're traveling tomorrow. Is that complete, or is there anything else?

Customer: That's it, yeah. I just need to know someone has it and can get it back to me.

At this point, emergence is complete. The full problem statement has surfaced: the time, location, phone status, emotional urgency, the failed self-call, the travel deadline, and the security concern. The solution statement can now be named: **the phone is in the customer's possession and they know it is secure, within a timeframe that allows pickup before tomorrow morning's departure.**

Notice what the AI did not do. It did not say "I'm transferring you to our lost and found department" after the first sentence. It did not say "Can I get your name and the date of your reservation?" before the customer had a chance to express the urgency and the travel deadline. It did not offer a resolution ("Let me check on that for you right now") before the full problem had emerged. Each of those moves would have been efficient and would have cut the emergence short. The travel deadline and the security concern would have remained unexpressed. The workflow that launched off the truncated problem statement would have aimed at a smaller target than the actual solution statement.

What a poorly designed system would have done:

Customer: Hi, I was at your restaurant last night and I think I left my phone there.

AI: I can help with that. Are you reporting a lost item?

Customer: Yes.

AI: What type of item did you lose?

Customer: My phone.

AI: I'll transfer you to our lost and found department.

This exchange is three turns. It is efficient. It produced a category (“lost phone”) and a routing decision (“lost and found”). What it did not produce: the time window, the location, the fact that the phone has already been called, the travel deadline, the security concern, or anything that would allow the next step to be genuinely responsive to the full problem. The customer was collapsed into a category at the first sentence. The emergence was destroyed.

Common Designer Mistake

Mistaking Method 1 for Intake Design

Intake design is the construction of structured sequences — forms, question lists, classification trees — that capture information at the beginning of an interaction. It is a legitimate discipline with real value in contexts where the problem type is already known. When a customer navigates to a specific support page and selects “report a lost item” before any conversation begins, intake design is appropriate: the problem type has been established by the customer’s navigation.

Method 1 is the emergence phase that precedes any such classification. It is the phase in which the problem type is not yet established — not because the customer is withholding it, but because the full shape of the problem has not yet surfaced, even in the customer’s own understanding.

The designer error is importing intake design logic into the emergence phase. The result is an AI that asks “What is the nature of your inquiry?” in the second sentence, offers a menu of problem types in the third sentence, and begins structured data capture in the fourth sentence. By the fifth sentence, the customer is filling out a form. They have been processed. They have not been heard.

The moment you add an intake form structure to Method 1, you have destroyed it. The form imposes categories before the human has finished arriving. Categories narrow what the human is allowed to express. Narrowing too early produces an incomplete problem statement. An incomplete problem statement produces a workflow aimed at a partial target. A workflow aimed at a partial target will, with regularity, fail to reach the actual solution.

This error is committed by well-intentioned designers who believe they are being efficient. Efficiency in Method 1 is measured by one metric: did the full problem statement emerge? Not: how quickly did

we classify the problem? Not: how few turns did the conversation take? The goal is not fewer words. The goal is fewer wrong workflows.

Early patience prevents late repair.

Design Practice

Identifying the Moment of Emergence

The following is a transcript of a founder attempting to explain an unclear business idea to an AI in an emergence session. Read it carefully. Your task is to identify the exact moment at which the full problem statement has emerged — the moment at which you have enough to name the solution statement, and no earlier.

Founder: So basically what we're building is — it's like a marketplace, but it's not really a marketplace, it's more of a matching platform. For consultants. Specifically for consultants who have expertise in one industry but want to apply it to a different industry.

AI: Interesting. Say more about that — what does the match look like when it works?

Founder: So like, imagine someone who spent fifteen years in automotive supply chain and now they want to consult for aerospace companies, because the problems are similar but the aerospace companies don't know that person exists. And the automotive person doesn't know how to reach aerospace buyers.

AI: So the connection that's missing is between the consultant's specific expertise and the industries that could use it but don't know about them. What happens when that connection doesn't exist?

Founder: The consultant keeps working in automotive because that's where they have their network. And the aerospace company keeps hiring the same consultants they've always used, even if those consultants aren't actually the best fit. Both sides are stuck with what they know.

AI: And when the connection works well — when the match is made — what does that look like? What changes?

Founder: The consultant gets into a new industry, builds a new book of business. The buyer gets a perspective they couldn't have gotten from their usual network. It's genuinely novel insight, not just a different face from the same pool.

AI: Got it. So this is about unlocking expertise that exists but isn't findable outside its home industry. Who decides when a match is good? Is there a quality signal, or is it purely introduction?

Founder: That's the part we haven't fully figured out yet. We have some ideas about vetting — like, the consultant's prior work would be verified, and there would be some kind of structured first conversation to establish relevance. But honestly, that whole layer is still in progress.

AI: That's useful to know. So the core of what you're building is the introduction layer, and the quality verification piece is still being designed. When you're trying to explain this to someone — an investor, a potential partner — where does the explanation usually break down?

Founder: They keep calling it a staffing agency. Or they think it's LinkedIn. Neither of those is right. It's not staffing because we're not placing people, we're facilitating relationships. And it's not LinkedIn because LinkedIn is based on who you already know. This is specifically about crossing industry lines that LinkedIn can't cross.

AI: So the thing you need them to understand is what it is not — it's not staffing, it's not a network — and what it distinctly is: a cross-industry expertise bridge. And right now, when you explain it, people land in the wrong category.

Founder: Exactly. I can get through a ten-minute conversation and they still think I'm building a staffing agency.

Your task:

First, identify the moment at which the full problem statement has emerged. Be specific — name the exchange or the sentence. It is not the first moment the founder mentions the product. It is the moment at which the full nature of the problem — including the problem with explaining it — has surfaced.

Second, name the solution statement. Do not name the product, the feature, the pitch, or the mechanism. Name the condition that would make the problem no longer a problem. In this case, the problem includes both the communication breakdown and the underlying business gap.

Third, identify the earliest point in the transcript at which a poorly designed system might have collapsed the emergence. What sentence would have triggered a premature classification? What would the classification have been? What would have been lost?

Resist naming anything beyond the solution statement. No workflow. No tool. No feature. Just the target.

Key Takeaways

Method 1 has exactly one output: the problem statement and its equivalent solution statement. No workflow is designed, no tool is selected, no routing logic is built in Method 1.

Emergence means allowing the full problem statement to come forward through live human speech before any classification, routing, or resolution is attempted. Every speaking human arrives unfinished. The full problem statement emerges through expression, not extraction.

The AI's natural capacity for holding ambiguity, reflecting without collapsing, and waiting patiently is the engine of the emergence phase. The designer's job is to preserve this capacity, not engineer it away

with premature classification prompts or structured intake sequences.

The solution statement names the condition that would make the problem no longer a problem. It is specific but not prescriptive about mechanism. It is the design target for all subsequent methods.

The equation is: problem statement = solution statement. The messy side and the resolved side are equivalent. The messy side is not wrong. The resolved side is the target.

Method 1 ends when the solution statement has been named — not when the problem has been classified, not when the customer has been routed, not when a record has been created. Just when the target has been identified.

Mistaking Method 1 for intake design is the most common and most damaging designer error in this method. The moment a structured form appears in the emergence phase, the emergence is over and a partial problem statement is all that remains.

Early patience prevents late repair.

Chapter 7: Method 2 — Hallucination Gives You the Ideal

Opening Thesis

Once you have the solution statement, you do not go to the client yet. You go to the AI first — and you let it hallucinate.

This requires a brief reorientation. Hallucination is commonly discussed as AI's primary failure mode: the AI generates fluent, confident, plausible content that is factually wrong, contextually inappropriate, or detached from the actual situation. In production, hallucination is false completion. It is the AI behaving as though it knows something it does not, and the consequences of that behavior can range from mildly unhelpful to seriously harmful.

But the same generative capacity that makes hallucination dangerous in production makes it valuable in discovery. In production, hallucination is false completion. In discovery, hallucination reveals the ideal. These are not compatible uses of the same AI in the same workflow — they require structurally different settings, different prompting disciplines, and a designer who knows clearly which mode they are in at every moment. Method 2 is the one point in the life cycle where hallucination is not just permitted but required.

What Hallucination Means in This Context

The professional name for what Method 2 produces is a blind ideal workflow simulation. The colloquial name — the one used in this book — is hallucination. The reason to use both terms is this: the professional name describes the method accurately and is appropriate in client-facing conversation. The colloquial name is memorable and captures the essential quality of what is happening: the AI is generating a complete, confident, best-class workflow without being grounded in any specific client's current reality.

The key word is blind. The AI does not know what systems the client has. It does not know that there is no database. It does not know that the manager is usually unavailable on weekend evenings. It does

not know that there is an informal workaround involving a staff member named Melissa at the gift shop counter who keeps a handwritten log of found items in a spiral notebook. It does not know any of this. And it must not know any of it. Not yet.

This deliberate blindness is the designer's most important act in Method 2. The AI, given a clear solution statement and no knowledge of the client's current reality, will generate the pattern of the ideal workflow: the steps that would be taken, the systems that would be consulted, the authority checkpoints that would be required, the notifications that would be triggered, the verification steps that would be needed. It will generate this confidently and fluently, because that is what it does. The designer does not need the AI to be right about this particular client. The designer needs the AI to show them the shape of the ideal. The client's reality comes in Method 3.

The Critical Constraint: Keep the AI Blind

The contamination error — introducing client-specific reality into the Method 2 simulation — is the most consequential mistake in this method. It is worth examining in detail, because it is committed by experienced designers as often as by beginners, and always for reasons that feel reasonable in the moment.

The contamination error sounds like this: “I know they don’t have a database, so let me mention that in the prompt so the simulation is realistic.” Or: “They have a really small staff, so I should tell the AI that so it doesn’t generate a workflow that requires five people.” Or: “The manager told me they absolutely cannot commit to a 24-hour callback policy, so I should factor that in.”

Each of these statements is technically accurate about the client. Each of them, if included in the Method 2 prompt, produces the same result: a to-be model that has adapted to the as-is reality before the as-is reality has been formally documented. The AI, told that there is no database, will generate a workflow that works without a database. The AI, told that the staff is small, will generate a workflow sized for a small staff. The AI, told that the manager cannot commit to a 24-hour callback, will generate a workflow that does not require a 24-hour callback.

The result is not a to-be model. It is a slightly improved as-is model. It looks like an ideal workflow because it has been generated with some rigor and presented with some structure. But it has been pre-adapted to the client's current limitations. The gaps between this model and the client's reality are small — which means the transformation being sold in Method 4 is small. The client has paid for incremental improvement rather than movement toward best-class outcomes.

Do not contaminate the to-be model too early. The as-is model is reality. It is not destiny. The to-be model is the ideal. It should be generated without reference to the current reality so that the gap between them can be clearly measured and clearly presented.

The Production/Discovery Distinction

Understanding why hallucination is valuable in Method 2 requires a clear grasp of the structural difference between AI in production and AI in discovery.

AI in production is operating in a live workflow. It is interacting with real customers, making real commitments, triggering real actions in real systems. In this context, hallucination — generating confident output that is not grounded in actual reality — is genuinely dangerous. If an AI in production tells a customer “We found your phone” when no phone has been found, that is a hallucination event. The customer has received false information. Trust has been damaged. A correction is required. The production environment has no tolerance for this.

AI in discovery — specifically in Method 2 — is operating in a simulation. It is not speaking to a real customer. It is not triggering real systems. It is generating a model of how the ideal workflow would run. In this context, confident generation without grounding in specific reality is exactly what is needed. The AI should not hedge. It should not say “Well, if they had a database, then maybe...” It should generate the full ideal workflow as though all the necessary systems, authority paths, staff behaviors, and records existed. The designer will evaluate that model, compare it to the as-is reality in Method 3, and use the gap to drive the transformation conversation in Method 4.

The designer who does not distinguish between these two uses of AI will either suppress hallucination in both contexts — producing a production environment that is safe but a discovery environment that is weak — or will allow hallucination in both contexts, which is genuinely dangerous in production. The discipline is to know which mode you are in, and to configure the AI accordingly.

The Demo Mode Prompt

The following prompt template is the standard Method 2 invocation. It is structured to establish all three critical conditions: the AI is in simulation mode, it has a clear solution target, and it is explicitly not being given information about the client’s current reality.

You are in demo mode. You are not speaking to a real customer. You are not bound by any specific client’s current process. Your task is to simulate the best-class workflow for resolving this situation.

The problem statement is: [insert full problem statement from Method 1]

The solution statement is: [insert solution statement from Method 1]

Let the human explain themselves naturally. Work toward the solution statement. Narrate the internal steps you would ideally take as if the proper systems, staff notifications, records, checks, and authority paths existed. The goal is to reveal the ideal workflow pattern, not to describe current reality.

Do not hedge based on what a typical business might or might not have. Do not adapt to assumed limitations. Generate the best-class workflow as though it could be fully supported.

Three elements of this prompt deserve attention.

First, the phrase “in demo mode” and “not speaking to a real customer” establishes that the AI is in a simulation context. This is important not just for what the AI generates but for how the designer reads the output. The simulation output is a model, not a specification. The designer treats it as a pattern to interrogate, not a plan to implement.

Second, including both the problem statement and the solution statement gives the AI a clear vector. The problem statement sets the starting condition. The solution statement sets the target. The AI generates the path between them without knowing the specific obstacles that exist in the client’s reality.

Third, the explicit instruction not to hedge and not to adapt to assumed limitations is the guard against a specific AI behavior: the tendency to preemptively calibrate its output to what seems operationally realistic. Without this instruction, the AI may spontaneously generate a workflow that assumes a small organization, limited staff, no sophisticated systems — not because it was told any of this, but because it is drawing on its training data’s sense of what a typical small business looks like. That spontaneous calibration is its own form of contamination, and the prompt must explicitly block it.

What the Output Looks Like

A well-executed Method 2 simulation produces a clear to-be model: a sequence of steps that represents the ideal workflow from problem statement to solution statement, with authority checkpoints, notification paths, verification steps, and escalation paths visible.

For the lost-phone case, a blind ideal simulation would produce something like the following model:

Step 1 — Receive and record the report. The customer’s report is received and a record is created immediately. The record includes: customer contact information, item description, time and location of last possession, time of report, and current status of the item (unknown).

Step 2 — Notify relevant staff. A structured notification goes to all relevant staff roles — front-of-house, back-of-house, hosts, busers — with the item description and the name of the reporting customer. The notification includes a request to check common found-item locations.

Step 3 — Check and log found items. All currently found items are checked against the report. If a match is identified, the item record is updated to “potential match — unverified.” If no match is found, the record remains open and is queued for follow-up when the prior evening’s staff are available.

Step 4 — Verify the match. A designated authority — a manager or supervisor — reviews the found item against the report and verifies the match. Verification may include asking the customer for ad-

ditional descriptive details (lock screen image, case color, identifying scratches). The item record is updated to “verified match” only after this step.

Step 5 — Notify the customer. Once a verified match exists, the customer is notified via their preferred contact method. The notification includes the verification status, the pickup location, the available hours, and any special instructions.

Step 6 — Enable retrieval. The customer is given a specific retrieval path: where to go, who to ask for, what identification to bring. The staff member who will handle the retrieval is notified.

Step 7 — Confirm resolution. After retrieval, the record is updated to “resolved” with the retrieval date and time. If the item is not claimed within a defined period, the record is moved to an escalation path with defined disposition options.

This model is the to-be model. Every step is visible. Every authority checkpoint is named. The designer now knows exactly what to ask the client in Method 3: Do you have a record system? Where do found items currently go? Who owns the verification step? Who is authorized to release an item? What is your current notification path to staff? What happens if no one claims the item?

The to-be model has become the interview guide.

Full Example: Blind Ideal Simulation for the Possible Double Charge

The following is a summary of a blind ideal simulation for the “possible double charge on a credit card” case. The problem statement is: “I think I may have been charged twice for the same purchase on my credit card. I’m not certain — it might be two different transactions — but it’s making me nervous about my account.” The solution statement is: financial security restored — the customer knows exactly what each charge represents, has confidence that their account is accurate, and knows that any error has been identified and corrected.

Step 1 — Receive and document the concern. The customer’s concern is received and a case record is created. The record includes the customer’s account identifier, the approximate transaction amount in question, the date range of concern, and the customer’s statement of uncertainty (possible duplicate, not confirmed). The record status is “under review.”

Step 2 — Retrieve and display the transaction detail. The relevant transactions are retrieved from the system of record and presented to the customer in a clear format: date, merchant, amount, authorization code. The customer is shown both (or all) transactions in question simultaneously.

Step 3 — Facilitate customer identification. The customer is asked to review each transaction and confirm whether they recognize it. This step does not presuppose that either transaction is an error — it is a fact-finding step. The customer’s identification response is recorded.

Step 4 — Classify the outcome. Based on the customer’s review, the case is classified as one of three types: (a) both transactions are recognized and distinct — no error, (b) one transaction is unrecognized — potential fraud or merchant error, or (c) one transaction appears to be a duplicate — merchant processing error.

Step 5 — Authority checkpoint: route by classification. Classification (a) is resolved at the customer-facing level: the customer receives a clear explanation of both transactions and the case is closed. Classifications (b) and (c) are escalated to a specialist who has authority to initiate a dispute, reverse a charge, or contact the merchant. The customer is notified of the escalation path and given a resolution timeline.

Step 6 — Communicate the outcome. Regardless of classification, the customer receives a clear statement of what was found, what action was taken (or not taken), and what the final status of their account is. If any action is still pending, a follow-up commitment with a specific date is established.

Step 7 — Confirm resolution. The case record is updated to “resolved” when the customer has confirmed that their account accurately reflects what they believe it should reflect. If the resolution involved a charge reversal or dispute, confirmation includes the reversal posting, not just the dispute initiation.

This to-be model has four features worth naming for the designer:

The model maintains the customer’s uncertainty throughout Steps 1 through 3 rather than immediately presuming fraud or error. This is important because the customer said “I’m not certain.” A model that immediately launches a dispute process would be misaligned with both the problem statement and the customer’s expressed state.

The model includes a classification step with three branches. This is normal for financial workflows and the designer should note it explicitly, because the discovery questions for Method 3 will need to probe which of these branches the client’s current process handles well and which it does not.

The authority checkpoint at Step 5 is explicit. Not all resolutions can be handled at the first-contact level. The to-be model names where the authority boundary is. Method 3 will ask: who currently has this authority? Is it defined? Is it exercised consistently?

The resolution confirmation in Step 7 is defined as the customer’s account accurately reflecting their expectations — not the dispute being filed, not the charge being identified, but the full cycle of financial security restored. This is the solution statement operationalized as a completion condition.

Common Designer Mistake

Contaminating the Simulation with Client Reality

The contamination error is committing the discovery interview before Method 2 is complete, or — more precisely — telling the AI about the client’s current limitations before running the simulation.

The error appears in many forms. The designer mentions, in the Method 2 prompt, that the client is a small independent restaurant with no POS integration and a single manager who handles everything. Or the designer, during the simulation, sees an output step that requires a dedicated customer notification system and mentally notes “they don’t have that” and asks the AI to modify the step accordingly. Or the designer runs the simulation and then edits the output to remove steps that seem “unrealistic” for the client before presenting it as the to-be model.

Each of these is the contamination error. And the error is always committed for the same reason: the designer feels responsible for being realistic. They have heard the client’s situation and they do not want to generate a model that will seem naive or impractical. They want to show up with something that feels achievable.

This is a reasonable instinct. It is wrong.

Realism belongs in Method 3. Method 2 must generate the ideal without the filter of the client’s current limitations. The reason is not theoretical — it is structural. The to-be model is the measuring stick. It is what the designer holds up in Method 4 and says: here is best-class. Here is where you are. Here is the gap. Now let’s decide how far you are willing and able to move.

If the measuring stick has been adapted to the client’s current reality before the measurement is taken, it will show a small gap — or no gap at all. The designer will have nothing to sell except incremental automation. The client will get a better version of what they already have. And in six months, when that client’s competitor deploys a workflow that actually serves the solution statement, the designer will be left explaining why they did not show the client the full picture when they had the chance.

Do not contaminate the to-be model too early. The ideal is the ideal. The client’s reality is the client’s reality. They are different things. Method 2 is where the ideal lives, uncontaminated.

Design Practice

Running a Blind Ideal Simulation for the Bad Restaurant Experience

The case: a customer had a bad experience at a restaurant. The specific problem, fully emerged: the customer celebrated a significant anniversary at the restaurant, the service was slow to the point of embarrassing them in front of their guests, the server appeared indifferent to the situation, and when they mentioned it to the host on the way out, they received a shrug. The problem statement: the customer had a meaningful occasion damaged by a service failure and received no acknowledgment from the restaurant. The solution statement: the experience is made right — the customer’s sense that this restaurant values their patronage and their occasion is restored, and they have received an acknowledgment that is proportionate to the harm.

Your task:

First, write a complete Method 2 demo mode prompt for this case. Include both the problem statement

and the solution statement. Include the explicit instruction to generate without hedging and without adapting to assumed limitations.

Second, generate (or simulate generating) the blind ideal simulation output. Build a to-be model with at minimum six steps. For each step, name: the action taken, who or what takes it, what record is created or updated, and whether an authority checkpoint is required.

Third, identify the authority checkpoints in your to-be model. Mark each step that requires a decision by someone with organizational authority — not just a front-line acknowledgment, but a decision that commits the organization to a specific resolution.

Fourth, review your to-be model and identify what discovery questions it generates. For each step in the to-be model, write one discovery question you would ask the client in Method 3. There should be at least one question per step, and most steps should generate two or three.

This exercise is not asking you to be realistic about the restaurant's current capabilities. The contamination error is available to you throughout this exercise. Notice when you feel the pull to make the model smaller or more conservative, and resist it.

Key Takeaways

Method 2 is a blind ideal workflow simulation. The AI is given the solution statement from Method 1 and asked to generate the best-class workflow for reaching it — without knowledge of the client's current reality.

The defining constraint of Method 2 is that the AI must be blind to the client's as-is process. Introducing client-specific limitations into the simulation contaminates the to-be model, adapting it to the current reality before the current reality has been formally documented. This produces a slightly improved as-is model rather than a genuine ideal, and it eliminates the gap that Method 4 depends on for its transformation argument.

Hallucination — the AI's capacity to generate fluent, confident, complete output without specific factual grounding — is the engine of Method 2. In production, hallucination is false completion. In discovery, hallucination reveals the ideal. The designer must know which mode they are in and configure the AI accordingly.

The output of Method 2 is a to-be model: a step-by-step representation of the ideal workflow from problem statement to solution statement, with authority checkpoints, notification paths, verification steps, and completion conditions visible.

The to-be model is the interview guide for Method 3. Every step in the to-be model generates discovery questions about the client's current reality. The designer enters the discovery phase knowing exactly what gaps to look for.

The demo mode prompt establishes simulation context, provides the solution target, and explicitly instructs the AI not to hedge or adapt to assumed limitations. All three elements are load-bearing.

Chapter 8: Method 3 — As-Is Discovery Gives You Reality

Opening Thesis

Only after the to-be model exists should the designer work with the client to understand the as-is model. This sequence is not a formality — it is structurally necessary. The to-be model, generated in Method 2 without knowledge of the client’s current reality, functions as the discovery interview’s scaffold. Without it, the designer enters the discovery session without a framework for what to ask. Without it, the client’s description of their current process fills the design space entirely, and what fills the design space becomes the design target. That is the commodity error, and it happens quietly and immediately the moment the designer sits down to interview without the to-be model in hand.

As-is discovery gives you reality. Not an idealized version of reality. Not the official version. The real operational state of the organization — the informal workarounds, the authority gaps, the staff dependencies, the things that break during the lunch rush, the thing that was supposed to be fixed two years ago and still has not been. The as-is model is reality, not destiny. What exists is what the designer is working from, not what the designer is confined to.

The To-Be Model as Interview Guide

The designer enters the discovery interview with the to-be model already in hand. This changes everything about how the interview proceeds.

Without the to-be model, the designer’s questions are generic: “Tell me about your current process for X.” The client answers. The designer takes notes. The designer eventually constructs a map of current operations and uses it as the design specification. The result is an AI workflow built to support the current process.

With the to-be model, the designer’s questions are targeted: “In the ideal workflow, there is a step where the found item is formally verified by an authority before the customer is notified. Does anyone currently own that verification step?” The client answers. The designer records not just what exists but what exists

relative to what should exist. The gap is visible at each step, not in aggregate at the end.

This is the methodological core of as-is discovery: the designer is not conducting a neutral interview to learn what the client does. The designer is conducting a structured interview to determine where the client's current operations diverge from the to-be model and by how much. The to-be model is the measuring stick. The discovery is the measurement.

This approach also changes the client's experience of the interview. When the designer asks questions grounded in an ideal model, the client understands that the designer has done preparatory work. They are not being interviewed by someone who showed up blank. They are being interviewed by someone who has already thought carefully about what the workflow could look like. This professional positioning matters for Method 4, where the designer will present the gap and sell the transformation. The credibility established in Method 3 carries into that conversation.

What to Gather

As-is discovery has six categories of information to gather. Each category corresponds to a type of gap that commonly appears between the to-be model and the current reality.

Official process. What the organization says it does. The documented procedure, the training manual instruction, the response script. This is the starting point but not the ending point of discovery. Official processes are frequently partial, outdated, or aspirational rather than operational.

Real process. What the organization actually does. The researcher's goal in this category is to find every place where the real process diverges from the official process. These divergences are not necessarily problems — they may represent adaptations that work well. But they are always important data, because an AI workflow that follows the official process but not the real process will be rejected by the staff who actually operate in the real process.

Informal workarounds. The adaptations that exist outside any official or even acknowledged process. The spiral notebook at the gift shop counter. The text thread where managers share lost-item updates. The custom in which the closing manager personally checks all the tables before leaving rather than relying on bus staff. These are often invisible to management and irreplaceable to operations. If the AI workflow eliminates them without replacing their function, the workflow will fail in ways that are difficult to diagnose.

Authority points. Who is authorized to make which decisions. Who can commit the organization to a resolution. Who can release a found item. Who can authorize a charge reversal. Who can tell a customer that an experience will be made right and in what form. These points are critical because the to-be model names authority checkpoints, and the discovery process must determine whether those checkpoints are currently staffed, defined, and exercised.

Systems of record. Where information about the situation is currently stored. The POS system. The reservation log. The handwritten sheet in the host stand. The manager's memory. These systems are

often informal, fragmented, and inconsistent. The discovery interview must map all of them — not just the official ones — because they represent both the data assets available to the AI workflow and the data liabilities it will need to work around.

Human dependencies. The staff members, roles, or individuals whose knowledge, behavior, or relationships are critical to the current process and whose departure would cause the process to fail. These dependencies are not organizational failures — they are organizational reality. But they represent risk, and the AI workflow designer who does not name them will build a workflow that assumes they will always be present.

Discovery vs. Judgment

As-is discovery is not a performance review. The designer is not evaluating the client’s current operations. The designer is gathering reality.

This distinction matters practically. If the client perceives the discovery interview as an evaluation, they will present a sanitized version of their current operations. They will describe the official process rather than the real process. They will omit the informal workarounds because those feel embarrassing. They will claim that authority is clearer than it is. The designer will produce an as-is map based on this sanitized account, and the gap analysis will underrepresent the actual distance between the to-be and the as-is.

The designer prevents this by framing the discovery explicitly and consistently as reality-gathering rather than judgment. “I’m not evaluating how you currently do this — I need to understand how it actually works so that whatever we design fits your real operations, not an idealized version of them. The more honestly you can describe what actually happens, the better the workflow we can build together.”

Every organization has gaps between their official process and their real process. That gap is not shameful. It is data.

Discovery Interview Structure

The discovery interview follows a consistent structural pattern: begin with the official process, then probe for the real process, then surface the informal workarounds, then map the authority structure, then identify the systems, then document the human dependencies. Within each category, the questions move from open to specific.

The general movement is from what the client believes is true about their operations to what is actually true, and from the stable conditions to the stress conditions. Stress conditions — what happens when the manager is not available, what happens during a rush, what happens when the system is down — often reveal the most important gaps.

The following question types are used throughout the discovery interview:

The official question: “What is your current process for [to-be step]?” This establishes the baseline. The client will usually describe the official process first.

The reality probe: “And what actually happens? When a server finds a phone on a table, what do they do with it right now?” This is the move from official to real. It should be asked without judgment and with genuine curiosity.

The workaround probe: “Is there anyone in particular who tends to handle this? Or a specific place items end up that isn’t official but everyone knows about?” This surfaces the informal structure.

The stress test: “What happens when the manager is at a different location? Or during Saturday dinner service when everyone is at capacity?” Stress testing reveals the brittle points — the steps that require specific conditions to function.

The authority question: “Who is actually authorized to release an item to a customer — or to tell them definitively that it was not found? Does that authorization need to come from management, or can any staff member handle it?” This names the authority structure.

The gap acknowledgment: “It sounds like there’s currently no formal notification process to staff when an item is reported. Is that right? How does word currently get out?” This makes the gap explicit without judgment.

Full Example: As-Is Discovery for a Restaurant’s Lost-Item Process

The following is a realistic discovery interview for a small independent restaurant. The designer has the to-be model from Method 2 in hand and uses it as the interview scaffold. The restaurant contact is the general manager.

Designer: Let’s start with the official process, as far as there is one. If a customer calls to report a lost phone, what is supposed to happen?

GM: Honestly, it’s pretty informal. There’s no written procedure. If a host or manager gets the call, they’ll usually take the person’s name and number and say they’ll check and call back.

Designer: And do they call back? What does “check” actually involve?

GM: It depends on who took the call and how busy we are. Sometimes they’ll walk the floor and check the obvious spots. Sometimes they’ll ask around. It’s not consistent.

Designer: Where do found items physically go right now? If a server finds a phone on a table, where does it end up?

GM: There's a shelf in the manager's office. But also sometimes things end up at the host stand, or behind the bar if the bartender found it. There's no single place.

Designer: Is there any record of what's there? A log, a list, anything?

GM: Not really. There was a binder at some point but nobody uses it.

Designer: So if a customer calls about a phone and someone goes to check, they're basically relying on memory and checking multiple spots?

GM: Yeah, that's accurate.

Designer: Let me ask about the verification step. In an ideal workflow, before you tell a customer "we found your phone," you'd want someone to actually confirm the item matches the description. Who currently has the authority to do that? Can any staff member tell a customer we found their item?

GM: Technically it should be a manager. But in practice, if a host is handling the call and they see a phone that matches, they'll probably just tell the customer.

Designer: What could go wrong with that?

GM: We could tell someone we have their phone, they come in, and it's actually someone else's, or we can't find it. That's happened.

Designer: I assumed as much. Let's talk about notification. In the to-be model, when an item is reported, all relevant staff get notified. Right now, how does word get out to staff that something is being looked for?

GM: It really doesn't. If the manager who took the call remembers to mention it at the start of a shift, they will. But there's no systematic way for the kitchen staff or bussers to know.

Designer: Who closes up at the end of the night? And what's their process for found items?

GM: The closing manager does a walkthrough. If they find anything they put it in the manager's office. But again, no record, no log.

Designer: What happens to unclaimed items after a week? Two weeks?

GM: That's never been defined. There's a box in the back with a bunch of stuff in it.

Designer: Last question: is there one person on your staff who tends to be the informal memory for this kind of thing? Someone who people ask when something has been lost or when a customer calls?

GM: Honestly, yeah — Maria at the host stand. She remembers almost everything. If anyone would know if a phone was found, it's her.

Gap Annotations Against the To-Be Model:

To-be Step 1 (Receive and record the report): **Gap.** No record system exists. Customer name and number may or may not be noted. No case record, no status tracking.

To-be Step 2 (Notify relevant staff): **Gap**. No notification path exists. Word may be passed verbally during a shift transition if the manager remembers. No systematic notification to all relevant staff.

To-be Step 3 (Check and log found items): **Gap**. Found items are stored in multiple locations with no inventory. Checking requires physical search of multiple spots. No log of what is on hand.

To-be Step 4 (Verify the match): **Gap**. Verification authority is theoretically managerial but is exercised inconsistently. Front-line staff may confirm matches without authorization.

To-be Step 5 (Notify the customer): **Partial**. Notification happens, but it is inconsistent and may communicate false positives (saying the item was found when verification has not occurred).

To-be Step 6 (Enable retrieval): **Gap**. No defined retrieval path. No staff designated to handle retrieval. No customer instruction about what to bring or who to ask for.

To-be Step 7 (Confirm resolution): **Gap**. No resolution record. Items remain in the office or in the back box indefinitely with no defined disposition.

Human dependency: **Maria at the host stand**. Her institutional memory substitutes for several formal steps. Her absence creates significant operational risk.

This annotated as-is map is the output of Method 3. It is not a judgment of the restaurant's management. It is a clear-eyed account of the distance between best-class and current reality. This distance is what the designer brings to Method 4.

Common Designer Mistake

Letting the As-Is Become the New To-Be

The most consequential as-is discovery error is allowing the client's description of their current process to become the design specification. This happens when the designer enters the discovery interview without the to-be model — or with the to-be model but without the discipline to hold it as the measuring stick.

The error has a natural logic. The client describes their current process. It has genuine logic to it — workarounds that developed for reasons, authority patterns that reflect real organizational constraints, systems that were chosen for real purposes. The designer, listening carefully, begins to understand why the process is the way it is. Empathy develops. The as-is feels reasonable. And then the designer thinks: "Well, if that's how they do it, let's just make AI do that."

This is the commodity error. A commodity AI provider automates the current process. A serious AI workflow designer sells movement from as-is toward to-be. The as-is is not the design specification. It is the departure point.

The to-be model must be held as the measuring stick throughout the discovery interview. Every time the client's description of the as-is feels like a complete answer — like the natural end point of the inquiry —

the designer must return to the to-be model and ask: does the current process meet this step? If not, that gap goes on the map. The gap stays on the map even if the client explains why it exists. Explanations do not close gaps. They contextualize them.

The as-is model is reality, not destiny.

Design Practice

Discovery Interview Guide for the Possible Double Charge

The following case is drawn from the to-be model generated in Method 2 for the possible double charge scenario. Your task is to write a discovery interview guide — at minimum 15 specific questions — that you would use with a client whose staff handles billing inquiries. The to-be model, recall, has seven steps: receive and document the concern, retrieve and display the transaction detail, facilitate customer identification, classify the outcome, route by classification, communicate the outcome, and confirm resolution.

Organize your questions into five categories:

Record systems. What systems exist? Where is transaction data stored? Who has access? Can staff retrieve a customer's full transaction history in real time, or is this a multi-step process? Are records linked across channels (phone call, online portal, in-person)? Is there a case record system for billing inquiries, or are inquiries handled without a record?

Authority paths. Who is authorized to initiate a dispute? Who can authorize a charge reversal? Is this authority defined by role or by individual? Are there dollar-amount thresholds above which a supervisor is required? Can the same staff member who receives the inquiry also resolve it, or does resolution always require escalation?

Notification paths. When a billing inquiry is escalated, how does the specialist know it is coming? Is there a handoff record, or does the customer have to re-explain? Is the customer notified when their inquiry moves from one level to the next? Who tells the customer what the resolution was?

Staff behavior. What do staff currently say when a customer reports a possible double charge? Is there a script or training guidance? What is the most common incorrect response? What is the most common staff confusion point? Is there a specific step in the current process where staff frequently deviate from the official procedure?

Failure modes. What is the most common complaint from customers about how billing inquiries are handled? What is the most common escalation trigger — why do customers typically ask for a supervisor? Are there patterns in which types of billing issues are resolved quickly and which drag on? What happens when the relevant system is unavailable?

For each question, note in parentheses which to-be model step it corresponds to. This connects your discovery interview directly to the gap analysis you will produce at the end of Method 3.

Key Takeaways

As-is discovery gives you reality — the actual operational state of the client’s organization, not the official version and not the ideal version.

The to-be model, generated in Method 2, functions as the interview scaffold. The designer enters discovery knowing exactly what gaps to look for, because each step in the to-be model generates a specific set of discovery questions.

Discovery is reality-gathering, not performance evaluation. Clients who perceive the interview as an evaluation will sanitize their answers. The designer’s framing must make clear that honest description of the real process — including workarounds, gaps, and inconsistencies — is what makes the design work.

The discovery interview moves from official to real, from stable to stressed. The most important operational information often lives in the stress conditions and the informal workarounds, not in the official process.

The output of Method 3 is a documented as-is model annotated with gap markers — every place where the current reality falls short of the to-be model. These gaps are the material the designer works with in Method 4.

The as-is model is reality, not destiny. Allowing the client’s description of their current process to become the design specification is the commodity error. The designer must hold the to-be model as the measuring stick throughout the discovery interview.

Human dependencies — the Maria at the host stand — must be named explicitly in the as-is model. They represent both operational value and organizational risk, and the AI workflow must account for them rather than assume they are permanent.

Chapter 9: Method 4 — To-Be Selling Gives You Transformation

Opening Thesis

Method 4 is where the AI workflow designer earns the right to be called a designer rather than a vendor. The gap between the to-be model and the as-is model is not a problem to apologize for. It is not evidence of a client's organizational failure. It is not an awkward finding to soften before the real pitch begins. The gap is the transformation you are selling. Everything the designer has done in the first three methods — the careful emergence, the disciplined hallucination, the rigorous discovery — was preparation for this conversation.

To-be selling gives you transformation. Not a product demonstration. Not a chatbot launch. Not an automation upgrade. A defined, committed movement from where the client's operations currently stand toward the best-class outcome the designer has already modeled. The designer arrives at Method 4 holding two things: a map of the ideal, and a map of the real. The distance between them is the value proposition. The path across that distance is the design.

The Commodity Provider vs. The Serious Designer

The difference between a commodity AI provider and a serious AI workflow designer becomes most visible in Method 4. It is worth stating this difference precisely, because it is often obscured by surface similarities: both parties are talking about AI with a client, both are showing demos or prototypes, both are discussing what the AI will do. The difference is in what they are selling and what they are promising the client will change.

The commodity provider's posture is: "Tell me your current process and I will make AI follow it." The premise is that the client's current process is the specification. The goal is automation — making the current workflow faster, cheaper, or more available. If the current process involves a staff member answering calls and directing them to a physical shelf in the back where lost items have been placed without a log, the commodity AI provider builds an AI that answers calls and directs customers to a physical shelf. The shelf still has no log. The AI just answers the call instead of a person.

The serious workflow designer's posture is different: "Here is the best-class to-be model for reaching the solution statement. Here is your as-is reality. Here is the gap. Now let's decide together how far toward the to-be model your business is willing and able to move, and what has to change in your operations to get there." The premise is that the to-be model is the measuring stick, the as-is is the starting point, and the designer's job is to sell the client on both the AI and the process changes that the AI's effectiveness requires.

A commodity AI provider automates the current process. A serious AI workflow designer sells movement from as-is toward to-be.

This distinction is not merely about quality or professionalism. It is about whether the client's situation actually improves. An AI that perfectly automates a broken process does not fix the process. It fixes the broken process in place and makes it harder to change, because now a technology dependency has been added to the organizational inertia. The client's satisfaction at deployment will turn to frustration when they realize that nothing fundamentally improved — the same problems recur, just with a different surface interface.

What the Designer Is Not Selling

Method 4 requires clarity about the product of the engagement. This clarity is not always comfortable, because what the designer is not selling is often what the client thinks they are buying.

The designer is not selling AI magic. There is no version of this conversation where the designer implies that deploying an AI will resolve a broken process by itself. The AI cannot verify a found item if no one has defined what verification means. The AI cannot notify a customer if the found item record does not exist. The AI cannot escalate to a manager if the authority path has not been established. The AI is a participant in the workflow, not the author of it.

The designer is not selling a voice agent. A voice agent is a delivery mechanism. It is how the customer interaction reaches the workflow. A well-designed workflow with a poorly-designed voice agent will underperform. A well-designed voice agent delivering a poorly-designed workflow will underperform. Selling the voice agent as the product confuses the mechanism with the outcome. The outcome is the solution statement. The mechanism serves the outcome.

The designer is not selling a prompt or a chatbot. These are implementation details. The client does not care about prompt engineering; they care about whether their customers' lost phones are returned, their billing disputes are resolved, and their guests' bad experiences are made right.

What the Designer Is Selling

The designer is selling movement. Specifically, movement from as-is toward to-be.

This is a more complex product than a software license. It requires the client to commit to changes that may be organizationally difficult: defining ownership where none existed, creating records where there were none, changing staff behavior, establishing authority paths, setting manager response expectations, building pickup protocols. These are not AI changes. They are business process changes. The AI workflow cannot run at its designed capability without them, and the designer must sell both.

The key question that anchors Method 4 is this: should we bend AI around the current process, or improve the current process so AI can deliver the to-be outcome? In most engagements, the answer involves some of both — there are places where the current process is reasonable and the AI simply needs to work within it, and there are places where the current process must change because the to-be model cannot be reached without the change.

The designer's job is to name which is which, to present the change requirements with clarity and without judgment, and to help the client understand that the process changes are not an inconvenient prerequisite for the technology. They are the primary intervention. The AI is how the improved process is delivered.

Business Process Changes the To-Be May Require

The gap analysis from Method 3 typically reveals a set of process changes that the client must make in order to support the AI workflow. These changes fall into predictable categories, and naming them clearly in the Method 4 conversation is part of the designer's responsibility.

Define ownership where none exists. If no one currently owns the lost-item process, someone must own it before the AI workflow is deployed. The AI can prompt for action, notify staff, and escalate — but it cannot perform the human steps that require a committed owner. The designer must name this gap and help the client designate ownership.

Create simple records where there are none. An AI workflow that requires a case record cannot function if no case record system exists. This does not necessarily mean a sophisticated database — a simple structured log may be sufficient. But something must exist, and the designer must help the client understand that the record is not a technology requirement. It is an operational requirement that the technology can support.

Change staff behavior. Staff who currently improvise will need to follow the process. This is a change management requirement, not a technology requirement. The AI can prompt the right steps. But if the staff do not take those steps, the workflow breaks at the human touchpoints. The designer must name this requirement explicitly, because it will determine whether the client needs training investment, communication planning, or operational supervision.

Establish authority paths. If the current authority structure is informal and inconsistent — anyone can release a found item, anyone can promise a resolution — the authority path must be defined before the AI workflow is deployed. The AI will hit authority checkpoints. If no one has been designated to fulfill them, the checkpoints will be ignored or bypassed, and the workflow will generate false resolutions.

Define status. What does “verified” mean exactly? What does “resolved” mean? What constitutes a completed retrieval? These definitions may seem obvious until the designer asks the client to define them precisely, at which point it often becomes clear that there is no shared definition. The AI workflow requires clear status definitions because it must know when to move from one state to the next. The designer must facilitate these definitions in Method 4.

Create pickup protocols. If a customer is told their item was found, how do they retrieve it? Who do they ask for? What identification do they need? What hours is retrieval available? These are process questions that the workflow requires answers to before it can close the loop. If the designer deploys the workflow before these questions are answered, the workflow will generate notifications that lead to frustrating customer experiences at the pickup step.

Define escalation paths. When the AI cannot resolve a situation, who is notified? By what method? Within what timeframe? What constitutes a failure of escalation? These paths must be defined, staffed, and tested before deployment.

How to Present the Gap Without Insulting the Client

The gap between the to-be model and the as-is model represents real shortcomings in the client’s current operations. The client knows this. The designer’s challenge is to present the gap in a way that is honest, specific, and actionable — without triggering the defensiveness that makes clients minimize the gap or resist the required changes.

The framing that works is discovery framing. The as-is was discovered, not diagnosed. “In our discovery, we found that the found-item record is informal” is different from “You have no record-keeping.” Both are true. Only one invites collaboration toward a solution.

The to-be model is framed as an aspiration, not a criticism. “Here is what best-class looks like” is a statement about the ideal. It is not a statement about the client’s failings. The client did not design an inferior process — they developed the process that was possible given their resources, attention, and priorities. The designer is offering them a path to a better one.

The gap is framed as an opportunity, because that is what it is. The gap is where the value lives. If there were no gap, there would be nothing to transform, and nothing significant to design. The designer should convey genuine interest in the gap — not as a problem to apologize for, but as the working material of the engagement.

The Transformation Target

The output of Method 4 is not the to-be model. The to-be model was generated in Method 2 and has been waiting through Method 3. The output of Method 4 is the transformation target: the agreed workflow the

client is willing and able to support, which is the best reachable position toward the to-be model given current resources, capacity, and organizational will.

The transformation target is not a compromise. It is a specific, achievable design destination that has been negotiated openly, with both the ideal and the current reality visible. It may be very close to the to-be model if the client is resourced and motivated. It may be farther away if the client has significant constraints. Either is legitimate — provided the designer has been honest about the gap and the client has made an informed choice about how far to move.

The workflow is not sacred. The solution statement is sacred. If the transformation target is the most the client can reach today, that is the design. But the solution statement does not change. The designer and the client are aligned on where they are going, even if the agreed path takes them there incrementally.

The To-Be Selling Conversation Structure

Method 4 follows a five-part structure in the client conversation.

Present the to-be model. Show the client the best-class workflow generated in Method 2. Walk through each step. Name the authority checkpoints, the notification paths, the verification steps, the completion conditions. Present it as what the ideal looks like — not as what is expected immediately, but as what is possible.

Present the as-is reality. Using the annotated gap map from Method 3, show the client their current reality against each step of the to-be model. Use the discovery framing — “In our discovery, we found...” — and present the gaps step by step without editorializing.

Name the gap. After presenting both models side by side, make the gap explicit: here is what best-class looks like, here is where you are, and here is what separates the two. Name each gap clearly. Do not bundle them into a vague summary. The client needs to understand each gap individually in order to commit to closing each one.

Present the transformation options. For each gap, the designer presents a specific change option: what process change is needed, what it would require from the client’s operations, and what the AI workflow can do in support of it. Some gaps may be closed quickly. Others may require a phased approach. Some may be deferred. The designer presents the options; the client decides.

Agree on the transformation target. The client makes explicit commitments to specific process changes that are prerequisites for specific workflow capabilities. The agreed workflow is documented with the process change commitments attached. This documentation is the design specification for the deployment.

Full Example: To-Be Selling Conversation for a Restaurant's Lost-Item and Complaint Workflow

The following is an excerpt from a realistic to-be selling conversation between a designer and the general manager of a restaurant. The designer has the to-be model from Method 2 and the as-is map from Method 3.

Designer: I want to walk you through what we found. I'll start with the to-be model — what best-class looks like for handling lost items and complaints — and then show you where things stand today. The goal of this conversation is to agree on a specific design target that your operations can actually support.

GM: Okay, let's see it.

Designer: In a best-class workflow, when a customer calls about a lost item, the first thing that happens is a case record is created immediately — customer contact, item description, time and location, report time. All of that. And that record stays open until resolution is confirmed. Right now, we found that there's no formal record system. Name and number may be noted, but there's no case tracking.

GM: I know. That's something we've talked about fixing.

Designer: Good — it's fixable, and it doesn't require a sophisticated system. A simple structured log, even digital, would meet the need. I'll come back to what that looks like. The second step is notifying relevant staff when an item is reported. In the to-be model, all relevant roles — hosts, servers, bussers, bar staff — get a notification with the description. Right now, word gets out informally, if at all.

GM: In practice, most of the notification is Maria. She's got a great memory.

Designer: She absolutely does. And that's a real asset. It's also a single point of failure. The workflow we design can't depend on one person being present. We need something that functions when Maria has a day off.

GM: Fair.

Designer: The third gap I want to highlight is verification. In the ideal workflow, a manager or supervisor verifies a match before the customer is told we have their item. In your current reality, front-line staff sometimes make that call, which has led to situations where you tell a customer you have their phone and it turns out to be a different phone, or you can't find it.

GM: That's happened. It's embarrassing.

Designer: It's also a trust issue with the customer. The AI workflow can hold the verification step — it will not notify the customer until a verified match exists. But that requires someone to be designated as the verification authority and to actually do the verification when the AI prompts them. Who would that be?

GM: It should be me, or the assistant manager.

Designer: Good. We need that to be explicit and it needs to be staffed — which means when you're unavailable, the assistant manager is the designated authority. The AI will notify you first and escalate to the assistant manager if there's no response within a defined window. Does that work?

GM: Yeah. What's the window?

Designer: That's for you to decide. Twenty minutes? Thirty? We need it to be fast enough that the customer gets a timely response, but realistic enough that it doesn't generate false escalations during busy service.

GM: Thirty minutes feels right.

Designer: Agreed. Let me now show you the complaint side. In the to-be model, when a customer contacts us about a bad experience, the workflow holds the full expression of the problem before responding. It then routes to either an immediate acknowledgment with a defined resolution offer, or an escalation to a manager for situations that require a commitment beyond a defined threshold...

The conversation continues in this pattern — presenting each gap, proposing the specific process change, and securing a commitment. By the end, the GM has agreed to: create a simple digital lost-item log, establish a staff notification protocol, define the verification authority, set a thirty-minute escalation window, and designate a resolution threshold for complaints below which front-line staff can offer a defined remedy.

These commitments are the transformation target. The designer's AI workflow is designed around them — not around the to-be ideal, and not around the as-is reality, but around the specific reachable position the client has committed to occupying.

Common Designer Mistake

Selling Only the AI Capability, Not the Process Transformation

The most common and most costly Method 4 error is selling the AI without selling the process change. The designer demonstrates the voice agent. The client is impressed. The call handling sounds professional. The responses are natural. The routing is smooth. The designer closes the sale.

Six months later, the client calls. Customers are still not getting their items back consistently. The AI is doing everything it was supposed to do. But the staff still do not log found items. The verification authority was never formally designated. The manager still handles escalations the same informal way as before. The workflow reaches its human touchpoints and breaks.

The client is dissatisfied. The designer is confused — the AI is working correctly. Both of these can be true simultaneously, and that is the problem.

The AI workflow worked perfectly against the client’s broken process. Nothing improved. The process transformation was not sold, not committed to, and not delivered. The client got a voice agent that follows their broken process with great fidelity.

This error is avoided by making the process changes explicit, commitments visible, and the connection between those commitments and the workflow’s effectiveness clear. “If the verification authority is not designated and active, the AI will generate a notification step that no one responds to, and the customer will never hear back about their item. This is not an AI failure. It is a process gap. We are closing that process gap together in this conversation.”

The client gets a working workflow when they commit to the process changes that the working workflow requires. Not before.

Design Practice

The To-Be Selling Dialogue for the Bad Restaurant Experience

The following exercise asks you to write the to-be selling conversation as a structured dialogue. You have the to-be model generated in the Method 2 design practice from Chapter 7, and the as-is map from your Method 3 discovery exercise. If you did not complete those exercises, construct a plausible as-is map based on what you would expect to find in a typical independent restaurant.

The case: a customer had a meaningful occasion damaged by a service failure — slow service, an indifferent server, and a dismissive response from the host on the way out. The solution statement: the experience is made right — the customer’s sense that the restaurant values their patronage and their occasion is restored, with an acknowledgment proportionate to the harm.

Write the to-be selling dialogue between the designer and the restaurant’s general manager. The dialogue should cover:

First, the to-be model presented in plain language — not jargon, not AI marketing language, but a clear description of what would happen in the ideal workflow when a customer contacts the restaurant after a bad experience.

Second, the as-is reality — what the designer found in discovery, presented with discovery framing (“In our discovery, we found...”) and without judgment.

Third, the gap — the specific distance between what the to-be model requires and what the current reality provides, step by step.

Fourth, the specific business process changes the designer recommends. There should be at minimum four, each stated concretely: who needs to do what differently, by when, and what the consequence is for the workflow if it is not done.

Fifth, the client’s hesitations. Write at least two points of client resistance — practical concerns about

staff capacity, doubt about whether customers will actually use the AI system, concern about cost or complexity. Write the designer's responses to each, grounded in the to-be model and the solution statement.

Sixth, the agreed transformation target. What specific commitments does the GM make at the end of the conversation? What does the designer commit to building against those commitments?

The dialogue should read as realistic conversation, not as a formal presentation. The client should ask questions, push back, and make reasonable concessions. The designer should hold the to-be model as the reference point throughout without being rigid about the path toward it.

Key Takeaways

Method 4 is where the gap between the to-be model and the as-is model is presented and sold as a transformation. The gap is not a problem to apologize for — it is the value proposition.

The serious AI workflow designer sells movement from as-is toward to-be. The commodity provider automates the current process. The distinction is whether the client's situation actually improves or merely gets a technology layer applied to the current reality.

The designer is not selling AI capability. The designer is selling a specific, achievable path from the current operational state toward the best-class outcome named in the solution statement. That path includes business process changes that are not AI changes: defining ownership, creating records, changing staff behavior, establishing authority paths, defining status, creating protocols.

The to-be selling conversation has five parts: present the to-be model, present the as-is reality, name the gap, present the transformation options, and agree on the transformation target.

The transformation target is not a compromise — it is an informed, negotiated, specific design destination that the client is willing and able to support. The workflow is designed against the transformation target, not against the to-be ideal.

The workflow is not sacred. The solution statement is sacred. The transformation target may reach the solution statement fully, or it may begin the movement toward it. Either is a legitimate design outcome, provided the solution statement remains the north star.

Process transformation commitments must be made explicit and documented before deployment. An AI workflow that is deployed against a client's broken process will perform correctly and achieve nothing.

Chapter 10: Method 5 — Stewardship Gives You Renewal

Opening Thesis

A workflow that works today may already be aging. This is not a metaphor. The AI models underlying a deployed workflow are updated — sometimes incrementally, sometimes substantially — on schedules that are outside the designer’s control. The client’s business evolves. Customer behavior shifts. New tool architectures become available. Cost profiles change. Voice quality improves. Reasoning capabilities expand. A workflow that was the best reachable design when it was deployed may, twelve months later, be significantly below what is now achievable. And if no one is asking the renewal questions, no one will know.

Stewardship gives you renewal. Not maintenance — renewal. Maintenance preserves the workflow. Stewardship renews the workflow around the solution statement. The distinction sounds subtle and is, in practice, fundamental. Maintenance asks: is the workflow still running? Stewardship asks: does the workflow still best serve the solution statement? These are different questions, and they require different disciplines, different cadences, and different organizational postures.

The Maintenance/Stewardship Distinction

Maintenance is the discipline of preserving an existing system. It is reactive: something has broken, and maintenance fixes it. It is conservative: the goal is to restore the system to its prior state. It is backward-looking: the reference point is how the system worked before the break. Maintenance is not a failure. It is necessary. But it is not sufficient.

Stewardship is the discipline of asking whether the existing system still earns its existence. It is both reactive and proactive: it responds to incidents, but it also schedules renewal reviews independent of whether anything has broken. It is forward-looking: the reference point is the solution statement, not the prior state of the workflow. Stewardship is willing to change a working workflow if a better one is possible. This is the hardest part of the discipline, and it is where most deployed AI workflows fail.

The failure mode looks like this: the workflow is deployed and performing within acceptable parameters. No incidents are reported. The client is broadly satisfied. The designer has moved on to other engagements. Eighteen months pass. The AI model underlying the workflow has been updated twice. The client has hired a new manager who handles escalations differently. A new tool architecture is available that would reduce workflow latency by forty percent. Voice quality in the new model generation is significantly better. The original prompt, written for a different model behavior profile, is now producing subtly different outputs than intended. None of these changes has triggered an incident. But the workflow that was best-in-class eighteen months ago is now meaningfully below what is possible.

There is an old software tradition that says: “If it isn’t broken, don’t fix it.” In AI workflow design, that tradition is not just wrong — it is actively harmful. A working workflow is not necessarily a good workflow. And a workflow that was good when it was designed is not necessarily good now. The steward’s job is to reject this tradition explicitly and replace it with a question: if we designed this today from scratch, would we design it the same way?

Two Kinds of Stewardship

Stewardship practice divides into two distinct modes. Both are necessary. Both require the designer’s active engagement. The first is incident stewardship — the response to a specific failure event. The second is renewal stewardship — the proactive review of whether the current workflow still best serves the solution statement.

Incident Stewardship

An incident is any event in which the workflow produced an outcome that was not aligned with the solution statement. The AI told a customer their phone had been found when it had not. The AI promised a resolution it could not deliver. A tool call failed silently and the customer was stuck. A prompt that worked correctly under the previous model version now produces outputs that are slightly but consequentially different. A manager notification was sent and ignored, and the escalation path was not completed.

Incident stewardship follows a clear discipline:

No blame, fast visibility. The first priority when an incident occurs is to see it clearly and quickly. Blame — whether directed at the AI, the designer, the client’s staff, or the model provider — interferes with visibility. The designer’s posture in an incident is neutral observation: what happened, what state the workflow was in when it happened, what the customer experienced, and what the intended outcome was.

Clean diagnosis. After visibility, diagnosis: why did the workflow produce this outcome? Was it a prompt failure — the AI interpreted an instruction differently than intended? Was it a tool failure — an API returned an unexpected state? Was it a model behavior change — the updated model handles a specific input pattern differently than the prior version? Was it a process gap — the workflow reached a

human touchpoint and the designated human did not fulfill their commitment? Clean diagnosis means identifying the actual cause, not the most convenient cause.

Contain quickly. Once the cause is identified, contain the incident: prevent it from recurring while the correction is being designed. Containment may mean temporarily routing specific input patterns to a human agent, disabling a specific tool call, or adding a fallback response while the underlying issue is resolved. Containment is not correction. This distinction matters. Containment is fast and conservative. Correction is deliberate and tested.

Correct thoughtfully. The correction addresses the root cause identified in the diagnosis. If the cause was a prompt failure, the correction changes the prompt — and tests the changed prompt against the range of inputs that produced the failure and the range of inputs that the prior prompt handled correctly. If the cause was a model behavior change, the correction may require rewriting the prompt to work with the new model's behavior profile, or it may require a deeper redesign. If the cause was a process gap, the correction may require returning to Method 4 — reopening the transformation conversation and securing new commitments from the client.

Renewal Stewardship

Renewal stewardship is not triggered by an incident. It is triggered by the steward's commitment to asking the renewal questions on a defined cadence, regardless of the workflow's apparent performance.

The renewal questions are:

Is this still the best way to get from the problem statement to the solution statement?

If we designed this today from scratch, would we design it the same way?

What has changed in the model — the underlying AI model's behavior, capabilities, reasoning depth, voice quality, tool use, or context window?

What has changed in the business — the client's staffing, operational structure, seasonal patterns, competitive environment, or customer expectations?

What has changed in customer behavior — the language customers use, the channels they prefer, the problems they are bringing, the expectations they have formed from other service experiences?

What has changed in available tools — new APIs, new integration options, new automation capabilities, new notification methods?

What has changed in cost, latency, voice quality, reasoning, or orchestration — factors that affect the workflow's performance and the client's economics?

These questions are not rhetorical. They require specific investigation. The steward reviews the model's release notes. The steward talks with the client about what has changed in their operations. The steward reviews interaction logs for pattern shifts in customer language or behavior. The steward checks the tool landscape for new options.

The output of a renewal review is one of two things: a decision to keep the current workflow because it still best serves the solution statement, confirmed by examination; or a renewed workflow that better serves the solution statement given what has changed. Either is valid. Inertia is not. A decision to keep the current workflow must be a decision, not an absence of questioning.

Be willing to obsolete your own design.

The Steward's Structural Position

The steward is not a support technician. The steward is not waiting for complaints. The steward is not managing a ticket queue. The steward holds a structural commitment to the solution statement and is authorized to ask — at any time and for any reason — whether the current workflow still best serves it.

This structural position is different from anything that exists in traditional software support. In traditional support, the metric is uptime and ticket resolution. The system either works or it does not, and support's job is to make it work. In AI workflow stewardship, the metric is alignment with the solution statement — which can degrade even when the system is technically operational and producing outputs within normal parameters.

The steward must be given real authority: the authority to recommend a workflow redesign, the authority to pause a workflow that is producing misaligned outcomes, the authority to reopen the Method 4 transformation conversation with the client. Without this authority, stewardship becomes a monitoring function — someone watching the logs but unable to act on what they see.

The renewal cadence should be scheduled explicitly, not left to the steward's initiative. Two triggering conditions should be built into the engagement agreement:

Scheduled renewal reviews: At minimum quarterly, and additionally at any major model upgrade event from the underlying AI provider. Model upgrade events are a specific trigger because they are the most likely source of silent drift in prompt behavior.

Incident-triggered reviews: Any incident that reveals a fundamental misalignment between the workflow and the solution statement — not just a tool failure — should trigger a renewal review, not just an incident correction. If the incident reveals that the workflow was reaching the human touchpoints and those touchpoints were consistently failing, that is not a prompt problem. It is a Method 4 problem, and the renewal review should reopen the transformation conversation.

Full Example: Lost-Phone Workflow Stewardship, One Year After Deployment

The restaurant deployed the lost-item and complaint AI workflow twelve months ago. The agreed transformation target at deployment included: a digital lost-item log, a staff notification protocol via a shared channel, a designated verification authority (the GM and the assistant manager, with a thirty-minute escalation window), and a defined complaint resolution threshold for front-line staff.

The following two scenarios show incident stewardship and renewal stewardship in operation.

Incident Stewardship: The False Positive

A customer calls to report a lost phone. The AI receives the report, creates a case record, and notifies staff. A staff member finds a phone in the manager's office that matches the description. The staff member updates the record to "potential match." The GM receives the verification prompt and responds "yes" — but the verification was done visually from across the room without holding the item, looking at the lock screen, or asking the customer for additional identifying details. The AI sends a notification to the customer: "We believe we found your item. You can pick it up at the host stand during business hours."

The customer arrives. The phone is the wrong model. The customer is told the phone cannot be located. The customer is upset — they were told definitively that it was found. They cancelled alternate arrangements based on that notification.

Incident diagnosis: The verification authority took the action but did not follow the verification protocol. The AI sent the notification based on the authority's "yes" response without any secondary confirmation. The prompt did not specify what "verification" required — it relied on the authority's judgment, which was exercised without sufficient rigor.

Containment: Temporarily route all verification-step notifications through a secondary confirmation — the authority must confirm both that the item matches and that verification was conducted with the item in hand before the customer notification is triggered.

Correction: Rewrite the verification prompt to specify the verification requirements explicitly: the item must be physically examined, the lock screen image or a specific identifying detail must be confirmed with the customer, and the authority must attest to these steps in their response. This is a prompt correction. It is tested against a range of verification scenarios before restoration.

Follow-up: The designer has a brief Method 4 conversation with the GM to restate the verification protocol and the consequences of a false positive. This is not blame. It is a process clarification triggered by a real incident.

Renewal Stewardship: The Twelve-Month Review

The steward schedules the twelve-month renewal review. The AI provider has released a major model version since deployment. The review follows the renewal questions.

What has changed in the model? The new model version has significantly improved voice naturalness and context retention across long conversations. It also handles ambiguous inputs differently — it is more likely to ask a clarifying question rather than make an inference. The original prompt was written for a model that would infer from partial information. The new model’s clarification behavior may be producing longer interactions than necessary for simple cases.

What has changed in the business? The restaurant has hired a second assistant manager and has opened a private dining room for events. The event space means that lost-item volume has increased, and a significant portion of lost items are now associated with event reservations rather than walk-in dining. The current case record does not have an event reservation field, which means event-related cases are harder to match and track.

What has changed in available tools? A new integration is available that would enable automated text notifications to customers when a verified match is found — eliminating the current manual step where the GM or assistant manager sends the notification. This integration was not available at deployment.

Renewal output: Three changes are proposed and agreed with the client.

First, the case record template is updated to include an event reservation field for event-associated cases.

Second, the prompt is revised to account for the new model’s clarification behavior — the revised prompt guides the model to be more directive in simple cases and reserve clarifying questions for genuinely ambiguous ones.

Third, the automated text notification tool is evaluated, tested, and added to the workflow, replacing the manual notification step. The GM’s involvement is now required only for the verification step, which is appropriate — that is an authority function, not a notification function.

The result is a renewed workflow that better serves the solution statement than the original deployed version did. The original worked. The renewed version works better. Neither the client nor the steward accepted the original as permanently adequate.

Common Designer Mistake

Treating Stewardship as a Post-Launch Support Contract

The support contract model is the wrong frame for Method 5. It produces a specific and consequential failure: the steward becomes a reactive function rather than a design discipline. The client calls with a complaint. The steward investigates. The steward fixes the complaint. The steward closes the ticket. The workflow returns to its prior state. No renewal question has been asked. No model change has been evaluated. No business evolution has been assessed. The workflow has been maintained, not stewarded.

This error is common because it mirrors the service contract structure that most clients understand and most designers have been trained to deliver. It is economically familiar: the client pays a monthly fee, incidents are resolved within a defined SLA, and the engagement continues until the client cancels. This

structure is not wrong as a commercial arrangement. It is wrong as a design discipline if it does not include the renewal function.

Stewardship is a design discipline, not a support ticket queue. The steward is not waiting for complaints. The steward is proactively asking the renewal questions. The steward is reading the model provider's release notes and evaluating whether behavior changes require prompt updates. The steward is checking with the client quarterly to understand what has changed in their operations. The steward is reviewing interaction logs for pattern shifts that are below the incident threshold but above the noise floor.

The steward who does only maintenance will eventually face a client whose workflow has drifted significantly from alignment with the solution statement, in ways that accumulated slowly and are now expensive to address. The early patience of regular renewal reviews prevents this late repair.

Early patience prevents late repair.

Design Practice

Stewardship Plan for the Sales Call to Decision Maker

The case: a founder with an unclear offer is using an AI-assisted workflow to support their outbound sales calls to decision makers. The AI workflow supports the call in three ways: it helps the founder arrive at the call with a clear, emergent problem statement from the customer's perspective (using a research and preparation phase); it assists in real-time by surfacing relevant proof points and reframing suggestions during the call; and it handles post-call follow-up sequencing based on the conversation outcome.

The solution statement: the decision maker has received an offer that is clear, relevant to their situation, and worth a second conversation — and the founder has a confirmed next step.

Design a stewardship plan for this workflow. Your plan should address the following.

Incident triggers to monitor. Name at least five specific events that should trigger an incident stewardship review. Be specific: not “the workflow fails” but “the AI surfaces a proof point that the founder later confirms was inaccurate” or “the post-call follow-up sequence sends a message to a decision maker who indicated they wanted no further contact.” Each trigger should be concrete enough that a steward could identify it from a log or a feedback report.

Renewal review questions. Write at least eight specific renewal questions for this workflow. The questions should be tailored to the specific context — the sales call workflow — not generic stewardship questions. Consider: what might change in how decision makers communicate their problems? What might change in the AI model's reasoning about offer clarity? What might change in the founder's business that would affect the offer itself? What might change in the competitive landscape that would require updating the proof points?

Renewal cadence. Specify a renewal cadence for this workflow. When should scheduled reviews occur?

What events beyond calendar schedule should trigger an unscheduled review? The cadence should be practical — something the designer and the client can realistically maintain — while being frequent enough to catch meaningful drift before it becomes expensive to address.

The renewal commitment statement. Write a brief paragraph — three to five sentences — that captures the stewardship commitment the designer makes to the client at the end of Method 4. This statement should make clear that stewardship is an ongoing design discipline, name the renewal cadence, and distinguish the designer’s stewardship role from a reactive support function.

Key Takeaways

Stewardship gives you renewal. Maintenance preserves the workflow. Stewardship renews the workflow around the solution statement. These are different disciplines with different questions, different cadences, and different relationships to the solution statement.

A working workflow is not necessarily a good workflow, and a good workflow at deployment is not necessarily good twelve months later. The AI models change. The business changes. Customer behavior changes. Available tools change. The steward’s job is to detect these changes and ask whether the current workflow still best serves the solution statement.

Two kinds of stewardship: incident stewardship responds to specific failure events with the discipline of no blame, fast visibility, clean diagnosis, contained quickly, and corrected thoughtfully. Renewal stewardship is proactive — scheduled and systematic — and asks the renewal questions regardless of the workflow’s apparent performance.

The renewal questions are specific and investigable: is this still the best path from problem statement to solution statement? If we designed this today from scratch, would we design it the same way? What has changed in the model, the business, customer behavior, and available tools?

Be willing to obsolete your own design. The steward who defends the current workflow out of investment in their prior work is not stewarding — they are guarding. Stewardship requires the intellectual honesty to recommend redesign when redesign is what the solution statement requires.

Stewardship is a design discipline, not a support ticket queue. The steward is proactively asking renewal questions, not waiting for complaints. The support contract model produces maintenance. Stewardship requires a different commitment, a different cadence, and a different authority.

Early patience prevents late repair. Regular renewal reviews prevent the slow drift that, undetected, produces an expensive realignment conversation with a frustrated client.

Method 5 feeds back into Method 1. After stewardship, the cycle renews. The problem statement may have evolved. The solution statement may have clarified. A new emergence may be required. The life cycle is recursive. The discipline is ongoing.

Part Three: Field Methods

From Framework to Practice

Part Two established the life cycle of an AI workflow: how a human problem emerges, how the ideal reveals itself, how reality is documented, how transformation is proposed, and how production sustains itself through stewardship. That is the map. Part Three is the territory.

A map can be studied at a desk. The territory must be entered. And when you enter it, you encounter things the map does not show — the hesitation in a customer’s voice before they get to the real problem, the workaround that one particular staff member has been running informally for three years, the authority path that exists in policy but has never been tested on a Tuesday evening. These are not failures of the framework. They are the texture of real practice, and they require methods, not just concepts.

Part Three presents six field methods in sequence. Each method corresponds to a phase of the workflow life cycle, but methods are not phases — they are practices you carry with you into specific working situations. You will not always apply them in linear order. You will sometimes run Method 2 before you have fully completed Method 1. You will sometimes discover during Method 4 that you need to return to Method 3. The sequence in this section is pedagogical, not prescriptive.

Chapter 11 opens with emergence sessions — the structured practice of allowing a human’s full problem to surface before any design decision is made. Chapter 12 follows with blind ideal workflow simulations, the counterintuitive method of running an AI through a scenario while deliberately withholding everything you know about the client’s current reality. Chapter 13 covers the as-is map, the reality document that shows what is actually happening inside an organization before any transformation is proposed. Chapter 14 addresses the selling conversation — how to present the gap between as-is and to-be in a way that opens a client to genuine process change rather than cosmetic AI adoption. Chapter 15 goes deep into the design decisions that determine whether a workflow is actually trustworthy: authority mapping, status language, and forbidden claims. Chapter 16 closes Part Three with stewardship and renewal — the ongoing practice that keeps a production workflow pointed at the solution statement as everything around it changes.

These six chapters will put you in the room. The concepts you carry from Part Two will tell you what you are looking for. The methods in Part Three will tell you how to look. # Chapter 11: Running Emergence Sessions

Opening Thesis

An emergence session is not a user interview. It is not a requirements gathering meeting. It is not a journey mapping exercise, a voice-of-the-customer survey, or a discovery sprint. It is the deliberate practice of allowing a human's full problem statement to come forward without premature collapse.

That distinction matters because almost every other research method in the design toolkit begins by imposing structure. A user interview has questions. A requirements session has a form. A journey map has predefined stages. Each of these imposes a category before the human has finished speaking. The emergence session begins from a different premise: the human arrives with a problem that is already fully formed, even if they cannot yet fully articulate it. The designer's job is not to extract it. The designer's job is to create conditions in which it can emerge on its own terms.

“The beginning of the workflow is not extraction. The beginning of the workflow is emergence.” This is not a philosophical preference. It is a practical claim about where design errors enter the system. Most broken AI workflows are not broken because the AI was poorly prompted. They are broken because the problem statement was never fully heard, and the design was built on a partial picture.

What an Emergence Session Is

An emergence session is a structured but deliberately open-ended conversation designed to surface the full problem statement — the complete articulation of what a human needs resolved, including dimensions they may not consciously recognize as part of the problem when they begin speaking.

The structure is held by the designer, not by the agenda. The designer knows what they are listening for. The human does not need to know the vocabulary of emergence to participate in one. The session can look to the human like a simple conversation. What makes it a session — rather than a casual exchange — is the designer's disciplined attention to what is emerging, what is being suppressed, and what has not yet been said.

“Every speaking human arrives unfinished.” This is the foundational assumption of the emergence session. No customer who calls about a problem has fully pre-formed their understanding of that problem before the call. The act of speaking — of being heard, of being asked a clarifying question — completes the articulation. A design method that does not account for this will consistently intercept the problem before it has fully arrived.

Session Formats

Emergence sessions can be conducted in several formats, each with different strengths.

Live call sessions place the designer in an active conversation with a real customer or a realistic proxy. The designer is not primarily asking questions. The designer is listening for emergence patterns — for the moment when the human's full problem statement begins to take shape. Live sessions allow real-time

annotation and the possibility of strategic non-intervention: the designer chooses when not to redirect, not to solve, and not to categorize.

Live transcript review sessions involve the designer observing an existing AI conversation in real time — watching a customer interact with a deployed AI workflow — while annotating the transcript as it builds. This format is useful when the designer cannot speak directly with customers but can observe the interaction. The risk in this format is that the workflow itself may be collapsing emergence prematurely, and the designer must distinguish between what the customer was trying to say and what the workflow allowed to be said.

Retrospective transcript analysis is the most common format in practice. The designer works with recorded transcripts — call recordings, chat logs, AI conversation histories — and annotates them after the fact. This format allows careful analysis but requires the designer to reconstruct the emergence pattern from a conversation that has already ended. The advantage is volume: a designer can analyze many transcripts and look for patterns across them.

Observed service interaction sessions place the designer in physical or virtual proximity to a real service interaction — a phone call, a counter exchange, an in-person complaint conversation — as a non-participating observer. The designer takes notes on emergence patterns without intervening. This format is the richest in context but the most logistically demanding to arrange.

The Observer Role

In every format, the designer's primary role is observation, not intervention. This is a discipline that conflicts with most professional training, which rewards decisive guidance, helpful redirection, and efficient question-asking. In an emergence session, these behaviors are often liabilities.

The designer who redirects a customer to the relevant department before the customer has finished explaining their situation has committed the most common emergence error: premature collapse. The designer who is “helpful” by suggesting solutions before the problem has fully formed has not been helpful — they have prevented the emergence session from doing its work.

The observer role requires the designer to track several things simultaneously:

What has the human said so far? What is the literal content of their statements?

What has the human not said yet? What dimensions of the problem are implied but not yet articulated?

Where is the emotional intensity? What words or moments carry charge — where does the human repeat themselves, speed up, slow down, or pause?

Where is the collapse risk? At what points is the AI, the human agent, or the system structure likely to impose a category before the human has finished emerging?

Where is the solution statement forming? What condition would make this problem no longer a problem?

Signs of Premature Collapse

Premature collapse is the moment when a system — an AI workflow, a human agent, a form, a menu — imposes a category before the full problem has emerged. It is one of the most frequent and most consequential errors in AI workflow design.

Collapse can be technical: an AI that has been prompted to branch on the first keyword it hears will route a customer before understanding their situation. If a customer says “I was charged,” the workflow branches to “billing dispute” before learning that the customer is actually calling because they want to understand a charge before deciding whether to dispute it — a very different problem statement with a very different solution statement.

Collapse can be conversational: a human agent who asks “Can I get your account number?” the moment a customer begins speaking has collapsed the conversation from a service interaction into a data-retrieval procedure. The customer’s problem may require the account number, but asking for it before understanding the problem tells the customer that the system is not interested in their problem — only in its own process.

Collapse can be structural: a phone tree, a web form, a chat widget that forces the customer to self-classify before reaching an agent collapses emergence at the intake point. The customer arrives at the conversation already categorized — often incorrectly — and the design must spend resources undoing a misclassification it caused.

“Do not collapse the human too early” is not a courtesy rule. It is a design principle with direct consequences for solution quality.

Transcript Annotation for Emergence Analysis

When conducting retrospective transcript analysis, the designer annotates the transcript systematically. A good annotation scheme marks the following:

First full problem statement (FPS): The moment in the transcript when the human’s complete problem statement becomes legible. This is often later than the designer expects. Many customers begin with a symptom (“I was charged twice”), move through context (“I only ordered one thing”), and arrive at the real problem (“I need to know whether I can trust this again before I use my card here”).

First premature collapse event (PCE): The first moment when the AI, the agent, or the system imposed a category before the FPS had been reached. Note what category was imposed, and what was lost by that imposition.

Emotional intensity markers (EIM): Words, phrases, or moments where the human’s language carries observable intensity — repetition, elevated register, explicit emotional labeling (“I was really frus-

trated”), or abrupt register shift (from polite to blunt). These markers often point toward the emotional center of the problem.

Repetition patterns (RP): When a human repeats a word, phrase, or idea across a conversation, they are usually signaling that this element is the emotional center of the problem — the thing they most need to be heard about. The designer highlights all repetitions and asks what they have in common.

Solution statement moment (SSM): The point in the transcript, if any, where the solution statement becomes legible. This may be explicit (“I just want to know it won’t happen again”) or implicit (the customer relaxes visibly or verbally once a particular assurance is given). Often the SSM is not in the transcript at all — the conversation ended before the human was fully heard, and the designer must infer the solution statement from context.

Identifying the Solution Statement from a Transcript

The solution statement is rarely stated explicitly. Customers do not call a service line and say “My solution statement is X.” They describe their problem, express their frustration, and occasionally hint at what would resolve it. The designer’s task is to infer the solution statement by asking a specific question: “What condition would make this problem no longer a problem?”

This question is not asked of the customer. It is asked by the designer of the transcript. The answer is often nested several layers below the surface complaint.

A customer who calls about a possible double charge is not, at the level of their full problem statement, asking about a refund. They may be asking whether they can trust the institution handling their money. They may be asking whether someone is paying attention. They may be asking whether making this call was worth their time. The refund is a step. Trust is the solution statement.

Avoiding Over-Intervention and Under-Constraint

There is a calibration problem in emergence session design that every designer must solve explicitly. If the AI is too constrained — given a detailed decision tree, heavy branching logic, early categorization triggers — emergence is impossible. The workflow will collapse the problem before it arrives. But if the AI is completely unconstrained in a production context, the conversation may drift, the customer may feel unmoored, and the interaction may never reach resolution.

The calibration point is specific: constraint should govern the resolution path, not the emergence path. The AI should be constrained in what it can commit to, what authority steps it can take, what states it can declare — but it should be relatively unconstrained in how long it listens before those constraints engage.

This is operationalized in the “do not help the AI” rule during emergence sessions. In a session designed to surface emergence patterns, the designer must not front-load the AI with such specific branching logic that emergence becomes impossible. Consider the difference between these two prompt approaches:

Kills emergence: “If the user says they lost something, ask for the item type, then the location where they believe they lost it, then their contact information.”

Preserves emergence: “A customer is reaching out about something they need resolved. Listen carefully to what they tell you. Let them describe their situation fully before you ask clarifying questions. Your first goal is to understand what they need, not to collect information.”

The first prompt commits the AI to a specific category (“lost something”) and a specific data collection sequence before the customer has said anything. The second prompt leaves the emergence path open while still orienting the AI toward resolution.

Running Emergence Sessions with Clients

When a designer conducts emergence sessions as part of a client engagement, they face a communication challenge: how to explain what they are doing without using vocabulary the client cannot interpret.

The explanation does not require technical vocabulary. A simple framing works: “Before we make any design decisions, I want to listen to how your customers actually talk to your AI — or to your staff — when they have a problem. I’m not analyzing the AI’s performance yet. I’m listening to the customer. I want to understand what they’re actually trying to say before we decide what the AI should do.”

Most clients respond positively to this framing because it is recognizable as good practice even without a theoretical vocabulary. The designer who can explain emergence sessions in plain language has a significant advantage over the designer who cannot explain their process at all.

Annotated Transcript Example: Possible Double Charge

The following is a realistic transcript of a customer calling about a possible double charge on their credit card. Annotations appear in brackets following each relevant segment.

Customer: Hi, yeah, I’m calling because I think I was charged twice for something? Or maybe — I’m not sure, actually. I’m looking at my statement right now and there are two charges from you guys, both for like twenty-three dollars? They’re on the same day.

[EIM: The customer arrives uncertain. “I think” and “or maybe” indicate the problem has not yet fully formed in the customer’s own mind. This is the emergence opening — the customer is still discovering their problem as they speak. Any categorization here is premature.]

AI: I can help you with that. Can I get your account number?

[PCE: First premature collapse event. The AI has moved immediately to data retrieval before understanding the problem. The customer is still mid-emergence. The account number request signals that the system is interested in its own process, not the customer's situation.]

Customer: Sure, it's... [gives account number]. And my name is Sandra.

AI: Thank you, Sandra. I'm pulling up your account. I can see two transactions on the fourteenth — one for \$23.18 and one for \$23.18. Can you tell me which merchant this is from?

Customer: It's from you guys. I mean, from — wait, which account is this? Is this my checking or my credit card?

[RP: The customer is confused about which account is being discussed. This confusion is important — it signals that the customer's real concern may be about account structure or oversight, not just about the specific charge. The designer notes this for later.]

AI: This is your checking account ending in 4421.

Customer: Okay. Yeah. So those two charges — I only remember buying one thing. I was at a restaurant, and I paid once. I don't know why there'd be two.

[EIM: "I only remember buying one thing" is the customer's attempt to establish their narrative. They are telling a story, not filing a form. The emotional weight here is around being wrongly charged — potentially being taken advantage of.]

AI: I understand. Would you like to dispute one of those charges?

[PCE: Second premature collapse event. The AI has moved to a resolution category — "dispute" — before the customer has reached their full problem statement. The customer may want to dispute the charge. But they may also want to understand what happened, verify whether this is a system error or a merchant error, and — most importantly — know whether this is a pattern. The dispute prompt cuts off this emergence.]

Customer: I mean — yes? Maybe? I just want to make sure I wasn't charged twice by mistake. Like, is this a thing that happens? Should I be worried about this?

[FPS: This is the first full problem statement. The customer is not primarily asking for a refund. They are asking: "Is my account safe? Is this institution paying attention? Should I be worried?" The solution statement is not "refund issued." The solution statement is closer to: "The customer understands what happened, is assured it is not a pattern or a security issue, and leaves the interaction with their trust in the institution intact."]

[SSM: The solution statement moment is visible in "Should I be worried about this?" This is the emotional center of the call.]

AI: Duplicate transactions can happen for a few different reasons. Would you like me to initiate a dispute for the second charge?

[The AI has not addressed “Should I be worried.” It has answered the narrower question with a procedural offer. From the customer’s perspective, this is another collapse — the system heard “charged twice” and offered “dispute process,” but did not hear the fear underneath the question.]

Annotation Summary:

- **FPS:** “I just want to make sure I wasn’t charged twice by mistake. Like, is this a thing that happens? Should I be worried about this?”
 - **PCE 1:** Account number request before problem is understood.
 - **PCE 2:** Dispute offer before full problem statement has been reached.
 - **EIM:** Customer uncertainty at opening; confusion about account identity.
 - **RP:** Pattern of “I only,” “I just want” — the customer repeatedly signals a minimal, reasonable ask that is not being met.
 - **Inferred solution statement:** The customer needs to understand what happened, receive assurance that their account is secure and the situation is not a pattern, and leave the interaction with trust in the institution restored. A refund, if warranted, is a component — not the goal.
-

Common Designer Mistake: Confusing Emergence Data with Requirements

The most common error designers make after running emergence sessions is treating the session data as a requirements document. A designer finishes analyzing three transcripts, writes up the findings, and labels the document “Customer Requirements.” They have already made a category error.

Emergence sessions give you problem statements and solution statements. They do not give you specifications. The distinction is consequential. A problem statement tells you what the human needs resolved. A solution statement tells you what condition would resolve it. Neither of these is a specification for how the AI workflow should behave, what data it should collect, what system integrations it needs, or what conversation flows it should follow.

A requirements document is the output of Methods 2 and 3 — the blind ideal simulation and the as-is map — translated into design decisions. A designer who skips from transcript analysis to specification writing has skipped the methods that reveal the ideal workflow and the current reality. They will build a workflow shaped by their interpretation of the transcripts, not by a disciplined understanding of the gap between what is ideal and what is real.

Emergence sessions are inputs to the design process, not its conclusion. Treat them accordingly.

Design Practice

Collect or write three realistic transcripts representing the following scenarios:

1. A customer calling a restaurant to complain about a bad dining experience.
2. A customer calling their bank about a possible double charge on their statement.
3. A business owner trying to explain their service offering to an AI that is helping them clarify their pitch.

For each transcript, conduct a full emergence annotation. Mark the following for each:

- **Emergence moment:** When does the customer’s full problem begin to form?
- **Collapse risk:** Where is the first point at which the AI or the system is likely to impose a category prematurely? What category is at risk of being imposed?
- **Emotional center:** What word, phrase, or moment carries the most emotional intensity? What is the customer actually worried about?
- **Full problem statement:** Write the full problem statement in one or two sentences, in your own words, after reading the full transcript.
- **Inferred solution statement:** What condition would make this problem no longer a problem? Write the solution statement in one sentence, beginning with “The [customer/founder/caller] needs...”

After completing all three annotations, compare the full problem statements across the three cases. Notice where they are similar and where they diverge. Consider: if you were designing three AI workflows — one for each case — how would the solution statements shape the design differently than the surface complaints would have?

Key Takeaways

Emergence sessions are the method by which the designer gives the full problem statement room to arrive. They are not interviews, requirements sessions, or journey maps. They are structured observations of a human problem in the act of becoming legible.

The designer’s role in an emergence session is primarily observational. The four things the designer tracks are: what has been said, what has not yet been said, where the emotional intensity lives, and where collapse is at risk.

Premature collapse is the most common and most consequential error in emergence session work. It can be technical, conversational, or structural — and it always costs the same thing: the full problem statement.

Transcript annotation is a specific skill. The designer marks the first full problem statement, the first premature collapse event, emotional intensity markers, repetition patterns, and the solution statement moment.

The solution statement is almost never stated explicitly. The designer infers it by asking: “What condition would make this problem no longer a problem?”

Emergence sessions produce problem statements and solution statements, not requirements. The designer who confuses the two has skipped the methods that bridge emergence to design. # Chapter 12: Running Blind Ideal Workflow Simulations

Opening Thesis

The blind ideal workflow simulation is the most counterintuitive move in AI workflow design. You already have the client. You already know their reality. You have sat with them, reviewed their processes, listened to their staff describe the informal workarounds and the gaps in the official policy. You know more about how this organization handles this problem than anyone outside the organization. And you must not tell the AI any of it.

This is not withholding for its own sake. It is not a methodological quirk. It is the precise mechanism by which Method 2 produces what Method 2 is designed to produce: the ideal workflow pattern, uncontaminated by the current reality. “Hallucination gives you the ideal.” The AI that does not know what the client cannot do will not adapt its suggestions to those limitations. It will describe what best-class resolution of this problem actually looks like — and that description is exactly what the designer needs.

“Do not contaminate the to-be model too early.” This is the governing principle of the blind ideal simulation. The to-be model must be built from the ideal, not from the as-is. The as-is will be documented in Method 3. The gap between them will be analyzed in Method 4. But at this stage, the designer’s job is to hold the ideal clear of the real, long enough to see what the ideal actually is.

The Simulation Setup

The blind ideal workflow simulation is conducted in a demo or sandbox environment. The AI must not be in a live production context, connected to real customer data, real system integrations, or real client configurations. If the AI has access to the client’s live systems, it will use them — and in doing so, it will replicate the as-is workflow rather than reveal the ideal.

The setup requirements are straightforward. The designer uses a clean instance of the AI — ideally the same model that will be used in the eventual production deployment, but isolated from any production configuration. There is no client-specific prompt in place. There are no API connections to client systems. There is no knowledge of the client’s staff, their informal workarounds, their known failure modes, or their existing process documentation.

The designer brings two things into the simulation: the problem statement and the solution statement, both derived from Method 1 emergence work. Everything else is deliberately excluded.

The Blind Constraint

The blind constraint is not just about data access. It is about the posture the AI brings to the simulation. An AI that has been told about the client’s current limitations will generate suggestions that work around those limitations. An AI that has been told that the restaurant has no lost-item log will not suggest referencing a lost-item log. An AI that has been told the client’s billing team is understaffed will not suggest an authority step that requires rapid billing team response.

This is the contamination problem. Even well-intentioned context — “by the way, they don’t currently have X” — shapes the simulation output toward the as-is. The designer ends up with a simulated workflow that is slightly better than the current reality but bounded by it. That is a very different output than the ideal.

The blind constraint also excludes what might be called the workaround inventory: all the informal knowledge about how things actually get done. Every organization has people like Melissa at the gift shop — the staff member who has been managing a category of problems informally because the official process doesn’t cover it. If the designer brings Melissa into the simulation (“assume there’s one person at the front desk who handles lost items informally”), the simulation will design around Melissa instead of revealing what the process would look like if it were properly designed.

Why This Produces Better Output

When an AI is given a problem statement, a solution statement, and no current-state constraints, it generates what it knows about best-class resolution of problems of this type. It draws on patterns across the full range of its training — service design patterns, hospitality industry best practices, resolution workflow patterns from industries the client may never have considered. It narrates what would ideally happen if every step were handled well.

This output is not a hallucination in the production sense. In production, hallucination is false completion — the AI claiming to have done something it has not done. But in the simulation context, the ideal output is precisely what the designer needs. “In production, hallucination is false completion. In discovery, hallucination reveals the ideal.”

The designer’s task is to read this output not as a description of reality but as a picture of what resolution could look like, stripped of the current state’s constraints. That picture becomes the to-be model. The to-be model then becomes the guide for the as-is discovery conversation in Method 3. And the gap between the to-be model and the as-is map becomes the transformation opportunity the designer brings to the client in Method 4.

The Full Demo Prompt Template

The following prompt template is designed for use in blind ideal simulations. It establishes the simulation context, enforces the blind constraint, orients the AI toward narrative output, and signals that the goal is pattern revelation rather than realistic performance.

You are in demo mode. You are not speaking to a real customer. You are not bound by any specific client’s current process. Your task is to simulate the best-class workflow for resolving this situation. The problem statement is: [paste problem statement]. The solution statement is: [paste solution statement]. Let the human explain themselves naturally. Work toward the solution statement. Narrate the internal steps you would ideally take as if the proper systems, staff notifications, records, checks, and authority paths existed. The goal is to reveal the ideal workflow pattern, not to describe current reality.

Several elements of this template deserve attention.

“You are in demo mode” is a framing signal. It tells the AI that the normal production constraints are suspended and the goal is exploratory. This matters because AI models trained for production use are often conservative — they hedge, they qualify, they avoid committing to steps they cannot verify. In simulation mode, the designer needs the AI to commit to the ideal steps, not hedge around them.

“You are not bound by any specific client’s current process” is the explicit blind constraint. It prevents the AI from importing any context it may have been given in earlier parts of the session.

“Narrate the internal steps you would ideally take as if the proper systems, staff notifications, records, checks, and authority paths existed” is the instruction that produces the richest output. It asks the AI to narrate what it would do if everything were properly set up — which reveals exactly the systems, notifications, records, and authority paths that the designer needs to document in the to-be model.

Reading the Simulation Output

The simulation output is typically narrative. The AI describes, in first person, what it would do at each step of the resolution process. The designer’s task is to convert this narrative into a structured to-be model.

This conversion requires the designer to read the narrative carefully for two types of content: pattern steps and authority steps.

Pattern steps are steps the AI can execute on its own — gathering information, confirming details, searching a record, sending a notification, updating a status, providing an explanation. These steps do not require external authorization to be real. The AI does them, and they are done.

Authority steps are steps where the AI narrates that it would “check with,” “wait for,” “notify,” “confirm with,” or “route to” a human actor, a system, or an external process. These are the steps where a state

change becomes real only when an external actor makes it real. The designer highlights every authority step in the simulation output and documents it in the to-be model as a handoff requiring design.

To-Be Model Structure

A well-formed to-be model derived from a simulation contains the following components:

Step list: The ordered sequence of steps from problem intake to solution statement achieved. Each step is named and described in one sentence.

Authority checkpoints: The subset of steps that require external authority to produce a real state change. For each checkpoint: who is authorized, what they authorize, how they signal authorization, and what triggers the request.

Pattern steps: The steps the AI handles without external authority. These form the AI's autonomous action sequence.

Status transitions: The set of honest status labels that correspond to meaningful state changes in the workflow. A status transition happens at each authority checkpoint.

Notification events: The moments in the workflow when a notification is sent to a customer, a staff member, a manager, or an external system.

Resolution condition: The specific state that constitutes achievement of the solution statement. This is the workflow's definition of done.

Variance Runs

A single simulation run produces one instance of the ideal workflow. But a well-formed to-be model should be stable across variations in the problem. If the workflow pattern changes dramatically when the problem statement is varied slightly, the to-be model is not yet well-formed — it is a response to one specific scenario rather than a generalizable workflow pattern.

The designer runs variance simulations by modifying the problem statement in controlled ways and observing whether the workflow pattern holds. For a restaurant complaint case, the designer might run the simulation with a mild complaint, a serious complaint, and a complaint involving multiple parties. If the ideal workflow pattern is genuinely stable, the core steps and authority checkpoints should be consistent across all three runs, even if the specific content at each step differs.

Variance runs also reveal the edge cases the workflow must handle. When the problem statement is pushed toward unusual territory, the simulation may reveal authority steps that were invisible in the standard case — a manager escalation, a regional oversight requirement, a legal hold scenario. These

edge-case authority steps are often the most important to design for, because they are the cases where the workflow is most at risk of false completion.

Extracting Authority Checkpoints from Simulation Output

The authority checkpoints are often the most important output of the simulation. They are the design artifact that most directly prevents the false completion problem. Extracting them requires the designer to read the simulation narrative with a specific question in mind: “At which steps does something become real only when a human or external system acts?”

Markers in the simulation narrative that indicate authority checkpoints:

- “I would then check with the manager on duty...”
- “At this point I would wait for confirmation from the billing team...”
- “I would notify the kitchen supervisor and wait for their acknowledgment...”
- “Once the transaction review has been completed by the finance team...”
- “I would hold the case open until staff confirms the item has been found...”

Each of these phrases signals that the AI has identified a step where its own action is insufficient to produce the real state. Someone else must act. The designer’s job is to name that someone, document what action is required, and decide what the AI tells the customer while that action is pending.

Full Example: Bad Restaurant Experience Simulation

Problem statement: A customer had a bad dining experience at a restaurant — slow service, an incorrect order, and a dismissive response from the server when they raised the concern. They are contacting the restaurant afterward to report the experience.

Solution statement: The customer needs to feel genuinely heard by someone with authority over the restaurant’s quality, receive a meaningful acknowledgment of the specific failures they experienced, and have confidence that their report will produce some real consequence within the restaurant — not just a form response.

Demo prompt:

You are in demo mode. You are not speaking to a real customer. You are not bound by any specific restaurant’s current process. Your task is to simulate the best-class workflow for resolving this situation. The problem statement is: A customer had a bad dining experience — slow service, an incorrect order, and a dismissive response from the server — and is contacting the restaurant afterward. The solution statement is: The customer needs to feel genuinely heard by someone with authority, receive a meaningful acknowledgment of the specific failures, and have confidence that their report will produce real consequences inside

the restaurant, not just a form response. Let the human explain themselves naturally. Work toward the solution statement. Narrate the internal steps you would ideally take as if the proper systems, staff notifications, records, checks, and authority paths existed. The goal is to reveal the ideal workflow pattern, not to describe current reality.

Simulated output (narrative form):

I would begin by giving the customer full space to describe what happened, without interrupting or redirecting. I would listen for the three elements of the experience: the service timeline, the order error, and the dismissal by the server. I would acknowledge each element specifically — not with a general apology but with a named acknowledgment of what went wrong at each point.

Once I had a full picture of the experience, I would inform the customer that I am creating a formal report of their visit, with the specific date, time, and party size, so that the general manager can review what happened. I would document the three specific complaints in the report: service delay, incorrect order, and dismissal response. I would not promise an outcome at this point — I would promise that the report reaches the right person and that the right person will respond.

I would then route the report to the general manager, flagging it as a post-visit guest experience complaint requiring manager review within 24 hours. The report would include the specific complaint details, the visit information, and the customer’s contact preference for follow-up.

I would notify the customer that the report has been submitted and that they will receive a personal response from the general manager or designee within one business day. I would provide a reference number for the report so the customer can follow up if they do not hear back.

If the customer asks what will happen to the server, I would explain honestly that their report becomes part of the staff feedback record and that the manager will review it in that context. I would not promise disciplinary action, because that is an authority-bound decision I cannot make. I would promise that the information will be seen.

The workflow closes when the manager has reviewed and acted on the report, and the customer has received a personal response. Resolution is confirmed when the customer acknowledges receiving the follow-up.

Extracted to-be model:

Step	Type	Status After
1. Customer describes experience	Pattern	Open
2. AI acknowledges each specific complaint	Pattern	Acknowledged
3. AI documents formal report with visit details	Pattern	Drafted
4. Report routed to general manager	Pattern	Submitted

Step	Type	Status After
5. Manager reviews report within 24 hours	Authority	Under Review
6. Manager contacts customer with personal response	Authority	Responded
7. Customer acknowledges receiving follow-up	Pattern	Resolved

Authority checkpoints identified:

- Step 5: General manager (or designee). Authorization: review of complaint report. Signal: manager action taken in management system. Trigger: report submission. Timeout: 24 hours; escalate to regional manager if not acted upon.
- Step 6: General manager. Authorization: content of personal response to customer. Signal: customer follow-up message sent. No AI-generated content for this step — human voice required to achieve solution statement.

Common Designer Mistake: Over-Specifying the Problem Statement

The simulation is sensitive to the specificity of the problem statement. A problem statement that is too specific produces a workflow that is only valid for one precise scenario. A problem statement that is appropriately general reveals the underlying workflow pattern that applies across the full range of relevant situations.

Consider the difference between these two problem statements for the restaurant complaint case:

Too specific: “A customer ordered the salmon and received the chicken instead, and when they told the server, the server said ‘that’s what you ordered’ and walked away. The customer is now calling the next day.”

Appropriately general: “A customer had a bad dining experience involving a service failure and an unsatisfactory response from staff, and is contacting the restaurant afterward.”

The too-specific problem statement produces a simulation that is precisely about salmon, chicken, and one specific server behavior. It may not generalize to a complaint about slow service with no order error, or a complaint about an atmosphere problem, or a complaint about a billing discrepancy. The designer who uses the too-specific problem statement will extract a to-be model that applies to a narrow slice of the actual problem space.

The appropriately general problem statement produces a to-be model that applies across the full range of post-dining complaints. The designer then runs variance simulations to test the model against specific sub-cases.

The goal is a to-be model at the right level of abstraction: general enough to cover the problem class, specific enough to be actionable. Most designers err toward over-specificity in their first simulation runs.

The correction is to ask: “Would this workflow apply to a variation of this problem? If not, what needs to change?”

Design Practice: Sales Call to Decision Maker

Run a blind ideal simulation for the following case: a salesperson is calling a decision maker at a prospective client company to qualify the opportunity and, if appropriate, advance the relationship.

Step 1: Define the solution statement.

The solution statement for this case is not “completed call” — that is a metric, not a resolution condition. Consider: what would make this situation no longer a problem for both parties? For the salesperson, the problem is unresolved opportunity. For the decision maker, the problem is unclear relevance.

A well-formed solution statement for this case might be: “The decision maker has a clear picture of whether this offering is relevant to their actual situation, and the salesperson has a qualified path forward — either a defined next step or a respectful disqualification that closes the loop for both parties.”

Note that “respectful disqualification” is explicitly part of the solution statement. A design that only works when the prospect is interested is not a full workflow design. The AI must handle both the qualified-forward path and the not-a-fit path with equal integrity.

Step 2: Write the demo prompt.

Using the template provided in this chapter, write the full demo prompt for this case. The problem statement should capture what is at stake for both parties. The solution statement should reflect the dual resolution condition above.

Step 3: Run or simulate the output.

If you have access to an AI in a sandbox environment, run the simulation and capture the full narrative output. If you are working through this exercise without a live AI, write the simulation output yourself — narrate, in first person as the AI, what best-class resolution of this scenario looks like step by step.

Step 4: Extract the to-be model.

From the narrative output, extract the step list, identify which steps are pattern steps and which are authority steps, document the authority checkpoints, write the status transitions, and define the resolution condition.

Step 5: Identify the key design tension.

In this workflow, the most significant authority step is the decision maker’s signal of interest or disqualification. The AI cannot produce that signal. It can only create the conditions in which the decision maker can give an honest signal. Consider: what does the AI’s role look like at that authority checkpoint? What is its job while waiting for that signal? What does it say if the signal is “not interested”?

Key Takeaways

The blind ideal simulation is the method that reveals the to-be model — the picture of what best-class resolution of the problem looks like, free of the current reality’s constraints.

The blind constraint is not a data security measure. It is a design necessity. An AI that knows the as-is will adapt to the as-is. An AI that does not know the as-is will reveal the ideal.

The full demo prompt template establishes simulation mode, enforces the blind constraint, and asks the AI to narrate internal steps as if proper systems existed. This produces the richest workflow output.

The to-be model extracted from a simulation contains: step list, authority checkpoints, pattern steps, status transitions, notification events, and resolution condition.

Variance runs test whether the workflow pattern is stable across variations of the problem. Stability indicates a well-formed to-be model. Instability indicates that the model is too narrow.

The most common simulation error is over-specifying the problem statement, which produces a to-be model that applies only to one precise scenario. The problem statement should be general enough to reveal the pattern, not so specific that it generates a single-case response. # Chapter 13: Building the As-Is Map

Opening Thesis

The as-is map is not a process audit. It is not a performance review, an efficiency analysis, or a compliance assessment. It is a reality document. Its purpose is not to judge the client’s current state — it is to make that state visible enough to transform.

This distinction determines how the designer enters the as-is discovery conversation, how they frame their questions, and how they handle the gap between what they observe and what they expected. A designer who approaches the as-is map as an auditor will find that clients become defensive, that staff minimize problems, and that the discovery conversation produces a sanitized version of reality rather than reality itself. A designer who approaches it as a documentarian — whose only job is to see clearly, not to evaluate — will find that clients open up, that staff describe the informal workarounds without embarrassment, and that the resulting map is rich enough to be genuinely useful.

“The as-is model is reality, not destiny.” This is the frame the designer carries into every as-is discovery conversation. The current state is what it is. It is not a permanent condition. It is a starting point. And to get from a starting point to a destination, you must first know exactly where the starting point is.

The To-Be Model as Interview Guide

By the time the designer enters the as-is discovery phase, they already have a to-be model. This is not an accident of sequencing — it is a deliberate method choice. The to-be model, built from the blind ideal simulation in Method 2, provides the designer with a complete picture of what the process would look like if it were working at its best. And that picture generates every question the designer needs to ask about the current reality.

Each step in the to-be model becomes an inquiry. If the to-be model includes a step called “Staff notified of lost item and asked to conduct physical search,” the as-is discovery question is: “When a customer reports a lost item, how do you currently notify staff?” If the to-be model includes an authority checkpoint called “Manager approves resolution action,” the discovery question is: “When a customer complaint reaches the point of requiring a resolution, who has the authority to decide what that resolution looks like, and how does that decision happen?”

This is not a leading interview technique. The designer is not asking these questions to steer the client toward the to-be model. The designer is asking them because they represent the complete set of workflow components that need to have as-is equivalents — or need to be documented as absent. The to-be model tells the designer exactly what to look for. The discovery conversation tells the designer what is actually there.

The Five Categories of As-Is Discovery

As-is discovery is organized around five categories. Each category reveals a different layer of organizational reality, and each layer is necessary to build a complete map.

1. Official Process

The official process is what the organization says it does — the policy document, the training manual, the onboarding script, the stated procedure. It may be posted on a wall. It may be embedded in a software system. It may exist only in the mind of a senior employee who has been “doing it this way for fifteen years.”

The official process is important to document because it is the organization’s stated intention. When the AI workflow is deployed, it will be expected to align with this stated intention — or the deployment will create visible conflicts that undermine staff confidence. The designer must know what the organization thinks the process is.

2. Real Process

The real process is what actually happens — what staff actually do when a situation arises, independent of what the policy says. In most organizations, the real process and the official process have diverged to some degree. Sometimes the divergence is trivial: a step is done in a slightly different order than

documented. Sometimes it is profound: an entire category of situations is handled through informal channels that have no official existence.

The real process is where the organization's actual operational knowledge lives. It has been refined by practice, shaped by past failures, and adapted to the real constraints of staffing, systems, and time. It is often significantly more effective than the official process at handling common situations — and significantly less effective at handling edge cases, because it has never been formally tested against them.

3. Informal Workarounds

Informal workarounds are the ad-hoc solutions that specific people have developed to bridge gaps between the official process and the real constraints of daily operation. They are almost always undocumented. They are often owned by a single person. They work because that person is skilled, attentive, and motivated — not because the process supports them.

The informal workaround is the most revealing artifact in the as-is map, because it precisely locates the gap between what the process provides and what the situation actually requires. Every workaround is a gap manifested. When the designer finds a workaround, they have found a place where the to-be model will need explicit design attention.

4. Authority Points

Authority points are the moments in the real process where a binding decision or real state change requires a specific human to act. Who is actually allowed to approve a refund? Who is actually allowed to tell a customer that their complaint has been escalated? Who is actually allowed to commit to a pickup time, a callback, a replacement, a credit?

These are not always the people listed in the official policy. In practice, authority often concentrates in specific individuals — the shift supervisor who is the only one anyone actually trusts to make calls, the back-of-house manager who handles all the situations the front-of-house policy doesn't cover. The designer must find the real authority points, because the to-be workflow will need to route to them.

5. Failure Modes

Failure modes are where the process breaks — especially under stress, volume, or the absence of key personnel. A process that works smoothly on a Tuesday afternoon may fail completely on a Friday dinner rush. A process that runs on one person's informal management may collapse entirely when that person calls in sick.

The failure modes reveal the fragility of the current state. They also reveal where the AI workflow will be most at risk of producing false completion — because the steps where the current process fails under stress are exactly the steps where the AI will need the most robust authority design.

Interview Structure and Technique

The as-is discovery interview follows a consistent structural arc. The designer opens with the official process, moves to the real process, probes for workarounds and authority points, and closes with failure mode testing.

Open with the official process question. “Walk me through how your team is supposed to handle [situation type].” This question establishes respect for the organization’s stated intention while signaling that the designer wants a full picture. It also gives the client something comfortable to answer, which opens the conversation before harder questions arrive.

Follow with the real process question. “Now walk me through what actually happens when this comes up.” Often the client will hesitate before answering this question. That hesitation is information. It signals that there is a gap between the official and the real. The designer waits through the hesitation.

Probe for the workaround person. “Is there one person on your team who kind of manages this informally? Someone people go to when something falls outside the normal process?” Almost every organization has this person. Naming them opens the workaround layer of discovery.

Test edge cases. “What happens during a Friday dinner rush?” “What happens when the manager isn’t there?” “What happens when you get three of these at once?” Edge-case questions reveal the fragility of the process under realistic operational stress. They also reveal the failure modes that will most severely test the AI workflow.

Test the failure floor. “Tell me about the last time this really went wrong.” This question often produces the most valuable data in the discovery conversation. The designer is not looking for blame. They are looking for a detailed example of what the failure mode actually looks like in practice — and specifically, at which step in the process the failure originated.

As-Is Map Components

A complete as-is map contains the following components for each process area being documented:

Process steps: The ordered sequence of steps as they actually occur in the real process, not the official process (though both should be documented when they differ).

Owners and roles: The specific person, role, or team responsible for each step. Where the official process names a role and the real process relies on a specific person, document both.

Systems used: The technology, tools, paper systems, or software in use at each step. Note where systems are absent and staff operate from memory or informal records.

Handoff points: The moments where responsibility passes from one person, role, or system to another. Handoffs are high-risk points for dropped context and false completion.

Informal paths: The workaround routes that specific people take when the official path is insufficient. These must be documented even though they are undocumented within the organization.

Authority checkpoints: The steps where a binding decision or real state change requires a specific authorized actor.

Common failure modes: The documented ways the process breaks — under volume, under staff absence, under unusual cases, under system failure.

Customer experience at each step: What the customer knows, what they have been told, and what they are experiencing at each step in the process. This is often the most sobering component of the as-is map — customers frequently know nothing about what is happening after their initial contact, because the internal process generates no customer-facing status signals.

Gap Annotation

For each step in the to-be model, the designer annotates the as-is map with a gap assessment. The four categories of gap are:

Equivalent present: The as-is has a step that accomplishes the same function as the to-be step, with comparable quality. Note what it is.

Partial equivalent: The as-is has a step that partially addresses the to-be function but is incomplete, unreliable, or person-dependent. Note the gap.

Workaround: The as-is has no official step but a specific person manages it informally. Note who, and note the risk of that person's absence.

Absent: The as-is has no equivalent at all — official, real, or workaround. This to-be step represents a capability the organization must build.

Gap annotation is the bridge between the to-be model and the as-is map. It tells the designer — and eventually the client — exactly what organizational changes are required to move from the current state toward the to-be state. It is the raw material for the transformation conversation in Method 4.

Document Structure

The as-is map should be formatted to allow direct comparison with the to-be model. The clearest format places the to-be step and the as-is equivalent on the same row, with the gap category and gap description in adjacent columns. This side-by-side format makes the transformation opportunity visible at a glance and provides the designer with a structured basis for the Method 4 selling conversation.

Full Example: Restaurant Lost-Item and Complaint Process

The following is an excerpt from an as-is discovery interview with the operations manager of a mid-sized restaurant group, followed by the resulting map with gap annotations.

Designer: Walk me through how your team is supposed to handle it when a customer calls after leaving and says they think they left something at the restaurant.

Operations Manager: The official procedure is that they call the main line, the host stand picks up, and the host checks the lost-and-found box at the front. We have a physical box near the host stand where anything found goes.

Designer: And what actually happens?

Operations Manager: *[pause]* Honestly, it depends who picks up. If it's one of the more experienced hosts, they'll check the box and if it's not there, they'll go ask the bussers. But a lot of times the call just gets transferred around. Sometimes it lands with the kitchen, sometimes they just say "nothing's here" without really checking. And the lost-and-found box doesn't always get updated — things get put in the back office or in a drawer somewhere.

Designer: Is there one person who kind of manages this more consistently?

Operations Manager: Yeah, there's one woman, Maria, who works Tuesday through Friday, and she actually keeps a handwritten notebook. She's been doing it for years. She writes down what was found, where, when, and the date. Nobody asked her to do that — she just started doing it. When customers call and Maria's not there, things fall through the cracks.

Designer: What happens when the customer calls and the item isn't there?

Operations Manager: We tell them it wasn't found. That's usually the end of it.

Designer: Do you do anything proactively to try to match found items to recent customers?

Operations Manager: No. There's no system for that. If something's found, it goes in the box or the back office. We wait for someone to call.

Designer: What happens on a Friday dinner rush if multiple people call?

Operations Manager: The host stand is swamped. The calls usually get a quick "let me check" and then either "not here" or "try calling back in the morning."

Designer: Tell me about the last time this really went wrong.

Operations Manager: A customer left prescription glasses. They called three times over two days. Each time they were told nothing was found. Third day they came in in person and the glasses were in a drawer behind the bar the whole time. They were pretty upset. Nobody had connected the calls. Nobody had really looked.

As-Is Map with Gap Annotations:

To-Be Step	As-Is Equivalent	Gap Category	Gap Notes
1. Customer contacts restaurant about lost item	Phone to host stand	Partial equivalent	Transfer risk; host stand may be occupied; no dedicated intake path
2. AI creates item report with description and contact	Maria's handwritten notebook	Workaround	Maria works Tue–Fri only; no official record system; no digital record
3. Staff conducts physical search of all locations	Host checks box; may ask bussers	Partial equivalent	Inconsistent; no defined search scope; back office not always checked
4. Match attempt against recent found-item records	None	Absent	No proactive matching; reactive only; no way to correlate across shifts
5. Customer notified of search status	Verbal on call only	Partial equivalent	No follow-up mechanism; customer must re-initiate contact
6. Manager notified if item not found after search	None	Absent	No escalation path; no manager awareness
7. Item held with identification tag if found	Ad-hoc	Partial equivalent	No tagging system; items may be misidentified in storage
8. Resolution status updated and customer contacted	None	Absent	No status tracking; no outbound contact if item is found later

Failure modes documented: - High volume: calls deprioritized at host stand; quick dismissals without genuine search - Key person absent: Maria not working; no notebook; no memory of prior calls - Multi-location storage: items dispersed across physical locations; no coordination - No match pathway: items found independently of calls; no mechanism to connect them

Common Designer Mistake: Accepting the Official Process as the Full Truth

The most consequential error in as-is discovery is stopping at the official process. The designer asks “how do you handle this?” the client describes the official procedure, and the designer writes it down and moves on. The resulting as-is map reflects the organization’s documented intention, not its operational reality.

This error is understandable. Clients often present their official process with confidence and some pride. Challenging it feels impolite. But the designer who accepts the official process as the full truth will build an AI workflow that fits the documented policy and conflicts with actual staff behavior. The staff will experience the new AI workflow as a foreign imposition that doesn’t account for how things actually work. They will develop new workarounds to manage the AI. The transformation will be nominal.

Every organization has a gap between policy and practice. In organizations that have been operating for several years without formal process review, that gap is often large. The real process is what runs the business — and it has been running the business precisely because it has adapted to the realities that the official process has not. Both must be documented.

The designer who completes an as-is map without finding a single informal workaround has not found an exceptionally well-managed organization. They have done an incomplete discovery.

Design Practice: As-Is Discovery for the Possible Double Charge Case

The following is a guide for conducting as-is discovery on a mid-sized regional bank’s phone support process for handling possible duplicate charge reports. Use the five discovery categories to organize your inquiry.

Official Process Discovery (Questions 1–6):

1. Walk me through the official policy for handling a customer who calls to report a possible duplicate charge on their account.
2. What documentation does a customer service representative create when this type of call comes in?
3. Is there a defined timeline for resolution of a reported duplicate charge? What is it?
4. What system does the representative use to look up transactions while the customer is on the phone?
5. What is the official escalation path if the representative cannot resolve the issue on the first call?
6. Does the official policy distinguish between a charge that has already settled versus one that is still pending?

Real Process Discovery (Questions 7–12):

7. In practice, what does a representative actually do first when this type of call comes in?

8. What does the representative tell the customer while they are looking up the account?
9. How long does a typical call of this type run? Does that match what policy suggests it should take?
10. When does a representative feel confident enough to tell a customer the duplicate has been confirmed — or denied?
11. Are there situations where a representative makes a judgment call that isn't in the policy? What does that look like?
12. How does the representative end the call if the transaction is still pending and no final determination can be made?

Informal Workaround Discovery (Questions 13–16):

13. Is there someone on the team who handles these calls differently — more effectively, or in a way that others have noticed works better?
14. Are there any shared team resources — spreadsheets, notes, group chats — that representatives use to handle these cases outside the official system?
15. What do representatives do when they can't find the answer in the system and the customer is waiting?
16. Is there a pattern of customers calling back about the same issue because the first call didn't fully resolve it?

Authority Point Discovery (Questions 17–19):

17. At what point does a representative need supervisor approval to take an action — for example, to apply a provisional credit or to officially open a dispute?
18. Who is the actual person a representative contacts when they need an authority decision? How do they reach them?
19. Can a representative commit to a timeline on behalf of the bank — for example, “your credit will appear within three business days”?

Failure Mode Discovery (Questions 20+):

20. Tell me about the last time a customer was very unhappy with how a duplicate charge call was handled. What happened?
21. What breaks down when call volume is high?
22. What happens when the transaction data in the system doesn't show a clear duplicate but the customer insists there is one?

Likely as-is gaps for a mid-sized regional bank's phone support:

The official process is likely well-documented at the intake level but incomplete at the resolution path level. Representatives are likely comfortable confirming what is visible in the transaction system but uncertain about how to handle cases where the data is ambiguous. Authority steps — especially around provisional credits — are likely under-defined, meaning representatives either over-commit (promising credits they cannot authorize) or under-commit (refusing to take action that is within their authority). The failure mode for this process is almost certainly the ambiguous-transaction case, where a settlement

is pending and neither the representative nor the customer has a clear status signal. That gap is where false completion risk is highest.

Key Takeaways

The as-is map is a reality document. Its purpose is not evaluation — it is visibility. A client who feels evaluated will produce a sanitized map. A client who feels seen will produce a complete one.

The to-be model serves as the interview guide for as-is discovery. Each to-be step generates a discovery question about what currently exists at that point in the real process.

The five categories of discovery — official process, real process, informal workarounds, authority points, and failure modes — must all be completed. Stopping at the official process produces an incomplete map.

The gap between official process and real process is not a problem to be fixed before the designer’s visit. It is one of the most important findings in the map. The designer who cannot find this gap has not yet completed discovery.

Gap annotation connects the as-is map to the to-be model. Each to-be step is assessed as having an equivalent, a partial equivalent, a workaround, or nothing in the current state. This four-category assessment is the raw material for the transformation conversation.

Informal workarounds locate the gaps precisely. Every workaround is a gap manifested. Finding the workarounds is finding the map’s most important data. # Chapter 14: Selling the To-Be Model

Opening Thesis

The to-be model is not a slide deck of AI features. It is not a demonstration of voice agent capabilities, a list of integration options, or a proposal for automating the current process. It is a picture of a human condition that has been resolved — and a design path for getting there reliably.

This is the distinction that separates serious AI workflow design from commodity AI deployment, and it must be made explicit in every to-be selling conversation. The commodity vendor shows the client what the AI can do. The workflow designer shows the client where the client’s customers need to arrive — and then shows the client what it will take to get them there. That is a fundamentally different offer. It requires a fundamentally different conversation.

“A commodity AI provider automates the current process. A serious AI workflow designer sells movement from as-is toward to-be.” The movement is the product. The AI is the mechanism by which some of that movement happens. But the movement requires process change — real, sometimes difficult, sometimes uncomfortable organizational change — and the designer who does not make this explicit is not actually selling transformation. They are selling comfort with a technology label on it.

What the Designer Is Selling

The designer is selling a direction. Not a product, not a technology, not a capability. A direction: movement from a documented current state toward a defined resolution condition, for a specific problem that real customers have.

This framing matters because it changes what the client must do to buy it. Buying an AI product requires a purchasing decision. Buying a direction requires a commitment — to process change, to organizational change, to staff change, to the willingness to build what the process currently lacks. The client who is not willing to make that commitment cannot actually buy what the designer is selling. The designer must understand this and must not paper over it.

The best clients are the ones who already sense the gap between their current state and the state their customers deserve. They may not have the vocabulary for it. They may not have the framework. But they feel it — in the complaint call they personally answered, in the staff turnover that correlates with customer friction, in the review that described exactly what they feared was true. The designer's job in the selling conversation is to make that felt gap legible and give the client a path through it.

The Gap Presentation

The gap presentation is the structured moment in the selling conversation when the designer places the to-be model, the as-is map, and the annotated gap in front of the client simultaneously. This is the core of the Method 4 conversation.

The framing matters enormously. Three framings must be in place before the gap is shown:

Frame the to-be as “what best-class looks like.” The to-be model is not the designer's opinion of what the client should do. It was generated by an AI in demo mode, working from the problem statement and solution statement without knowledge of any specific client's limitations. It represents what best-class resolution of this problem looks like across the industry. The client is seeing a standard, not a critique.

Frame the as-is as “what we found in the reality document.” The as-is map is not an assessment of the client's competence or intentions. It is a documentation of what was observed. The language should be precise and non-evaluative: “We found that notifications are handled informally by one staff member on certain shifts” rather than “Your notification process is inadequate.” The reality document is a mirror, not a report card.

Frame the gap as “the transformation opportunity.” The distance between the as-is and the to-be is not a deficit to be embarrassed about. It is the space in which the designer's work creates value. Large gaps represent large transformation opportunities. The designer who frames the gap as failure will make the client defensive. The designer who frames the gap as opportunity will make the client curious.

The Key Question in the Room

Every to-be selling conversation contains one pivotal question. It is rarely asked explicitly, but it is always present. The designer must make it explicit:

“Do we bend the AI around the current process — or do we improve the current process so the AI can deliver the to-be outcome?”

This question is not rhetorical. Both answers are possible. Bending the AI around the current process is technically feasible. It will produce a workflow that does what the current process does, faster and more consistently. It will also produce broken outcomes at scale if the current process is broken — because the AI will replicate the failure modes of the current process with greater efficiency and less human judgment.

Improving the current process requires organizational change. It requires staff to do things differently, to own things they haven't owned, to respond to signals they haven't been responding to. It is harder. It is slower. And it is where the transformation actually happens.

The serious designer makes this choice explicit in the room and holds the conversation there until the client has genuinely engaged with it. A client who has not consciously chosen a direction will default to the path of least resistance — which is almost always “can't the AI just handle it?” The designer must be prepared to answer this question directly.

Why Most of the Value Is in Process Change, Not AI Capability

This is the sentence that separates sophisticated AI workflow design from the broader market of AI deployment services: the AI can follow any process. If the process is broken, the AI will follow the broken process faster, more consistently, and at greater scale than any human workforce. The result is not better outcomes — it is broken outcomes delivered reliably.

The transformation value lives in the process change that the AI deployment enables. When the designer shows a client that their lost-item process has no notification system, no search protocol, no status tracking, and no escalation path — and then shows them what those things would look like if they existed — the transformation opportunity is in building those things. The AI is the reason to build them. But the AI does not build them. The organization builds them. The designer shows the organization why and how.

A client who deploys an AI voice agent over an unchanged broken process has purchased the acceleration of their current failure mode. The designer who allows this has not served the client. They have taken their money.

Process Changes to Propose

When the gap analysis reveals that the to-be model requires capabilities the as-is does not have, the designer recommends specific process changes. Each recommendation must be accompanied by a justification — a clear explanation of what breaks if the change is not made.

Define ownership. Every authority step in the to-be model requires a named owner — a specific person or role who is responsible for that step and who will be held accountable for acting within the required timeline. If ownership is not defined, the authority step will be ignored under load. The AI cannot act in place of an absent owner. It can only hold the case open until an owner acts — or declare false completion if no owner is assigned.

Create records. The AI cannot reference what does not exist. If the to-be model includes a step where the AI checks a lost-item log, there must be a lost-item log. If the to-be model includes a step where the AI routes a complaint to the manager, there must be a record system that accepts and holds complaints. The designer must identify every step where the AI needs to reference or create a record and ensure that the record system exists.

Change staff behavior. The AI cannot alert someone who will not respond. An authority step that routes to a staff member who ignores alerts — or who is present only on certain shifts, or who treats the alert channel as optional — will fail at exactly the moment it matters most. The designer must identify where staff behavior changes are required and be direct about what those changes are.

Establish authority paths. Someone must be authorized to act on each authority-bound step. The authority path must be defined clearly enough that the AI can route to it reliably, that staff know who to route to when the primary authority actor is unavailable, and that the escalation path functions under realistic conditions including the absence of key personnel.

Define status. Vague status creates false completion risk. If the to-be model includes a step where the case status moves from “submitted” to “verified,” there must be a defined action that produces the “verified” status — not an assumption, not an informal signal, but a real state change that the AI can detect or that a human actor explicitly records. The designer must work through every status transition in the to-be model and confirm that each one has a real trigger.

Create protocols. The AI cannot promise a pickup if there is no pickup protocol. It cannot promise a callback if there is no callback system. Every commitment the AI makes in the to-be workflow must be backed by an actual process commitment that the organization can execute. The designer audits every AI statement in the to-be workflow for commitment words — “will,” “you will receive,” “someone will call,” “we will hold” — and ensures each one is backed by a real protocol.

How to Handle Client Resistance

Resistance to process change is normal and should be anticipated. The designer who is surprised by resistance has not prepared adequately for the selling conversation. Each form of resistance has a specific honest response.

“We don’t have a system for that.” This is not a reason the change cannot happen. It is the first item on the implementation list. The honest response: “That is the first thing we build. If we do not build it, the AI cannot reference it, which means this step in the workflow cannot be real. We can design the minimum viable version of that system together.”

“Our staff won’t do that.” This is a real constraint that must be taken seriously and not dismissed. But it is not a permanent condition — it is a change management challenge. The honest response: “What would it take to change that? Is this important enough to the business to invest in the staff change? Because without this change, we are designing for a gap that will produce false completion at exactly the cases that matter most.”

“Can’t the AI just handle it?” This is the most common form of resistance, and it requires the clearest answer. The AI can handle the pattern steps. Every authority step requires a human to act. If the organization wants to promise outcomes that require authority-bound actions — resolutions, credits, found items, accepted agreements — then someone must be authorized to take those actions, and the AI cannot replace that someone. The honest response: “The AI can do everything that doesn’t require authority to be real. For the steps that do require authority, the AI’s job is to route to the right person and hold the case open honestly until that person acts. If we remove the authority steps, we get faster false completion — not better outcomes.”

The Transformation Target

The to-be selling conversation does not end with the client committing to the full to-be model. In most cases, the full to-be model is an aspiration for a longer-term implementation. The near-term commitment is the transformation target: the agreed workflow that is neither the pure to-be ideal nor the raw as-is, but the best reachable state the client is willing and able to support in the near term.

The transformation target has two requirements. First, it must be genuinely better than the as-is — not cosmetically different, not AI-branded, but meaningfully closer to the to-be along the dimensions that matter most for the solution statement. Second, it must be achievable given the real constraints of the client’s organization, including their willingness to make the process changes the transformation requires.

The designer’s job is to hold the line between an honest transformation target and a capitulation. Accepting a transformation target that is essentially the as-is with AI added is not a transformation. It is a deployment, and it is likely to produce the same failure modes as the current process, now backed by the credibility of an AI vendor.

Full Example: Selling the To-Be Model to a Restaurant Chain

The following is a condensed version of the to-be selling conversation for a restaurant chain that is considering deploying an AI voice agent to handle post-dining complaints.

Setting: The designer, the chain’s VP of Operations, and the Director of Guest Experience are in the room. The designer has already presented the to-be model (derived from a blind ideal simulation) and the as-is map (derived from discovery interviews with three restaurant managers). The gap annotation shows significant absences in the as-is.

Designer: The to-be model has seven steps from intake to resolved. The as-is has a partial equivalent for two of them and nothing for the other five. The five absent steps are: formal report creation, manager routing, 24-hour manager response, personal follow-up to the guest, and case closure confirmation. I want to walk through what that gap means before we talk about what to deploy.

VP of Operations: Right. So the AI would fill those gaps?

Designer: The AI can fill the pattern steps. Steps one, two, three, and four in the to-be — intake, acknowledgment, report creation, and routing — are all pattern steps. The AI handles those. Steps five and six require a manager to act: review the report and personally respond to the guest. Those are authority steps. The AI cannot do them. What the AI can do is hold the case open, track whether the manager has acted, and escalate if they haven’t.

Director of Guest Experience: We have managers. They respond to things.

Designer: I believe that. The question the as-is discovery raised is: do they have a consistent, visible path for responding to these complaints? Right now the guest contacts the restaurant, the call goes to the host stand, and the resolution depends entirely on who picks up and what they decide to do. There is no formal report. There is no routing to the manager. There is no defined response timeline. The AI can create all of that — but only if the manager actually responds when the routing arrives.

VP of Operations: What happens if the manager doesn’t respond?

Designer: Right now, nothing. The guest doesn’t hear back. The complaint disappears. With the AI in place, we can design an escalation: if the manager hasn’t responded in 24 hours, the case escalates to the regional manager. We can also give the guest a reference number and a commitment to a specific response timeline — which means the guest can follow up if they don’t hear back. That creates accountability the current process doesn’t have.

Director of Guest Experience: I like that. Can we start with that version?

Designer: Yes. But I want to be clear about what that requires from the operations side. We need: a defined escalation path for each restaurant, a manager response channel that generates a visible record, a

response timeline commitment — I'd recommend 24 hours — and a protocol for the manager's personal response to the guest. Those are not AI configurations. Those are organizational commitments. If we deploy the AI before those commitments are in place, the AI will take in complaints and route them to managers who aren't set up to receive and act on them. We will get faster abandonment, not better resolution.

VP of Operations: How long does it take to stand up those commitments?

Designer: The escalation path and the response timeline can be defined in one working session. The manager response channel needs a decision — is it email, is it the existing reservation system, is it something else? The personal response protocol is a training question. I can give you a two-page protocol draft as part of the implementation package. Realistically, four to six weeks to establish the infrastructure before we go live.

Director of Guest Experience: That seems like a lot before the AI is even doing anything.

Designer: I understand that feeling. But consider the alternative: if we go live before the authority steps are set up, the AI will handle the intake perfectly, create a beautiful report, route it into a void, and the guest will never hear back. The guest experience outcome is the same as today — or worse, because now the guest has a reference number and a commitment that nobody will honor. The four-to-six weeks is the work that makes the AI's output real.

VP of Operations: Okay. We'll commit to that timeline. What's our transformation target?

Designer: I'd recommend we target the following for launch: full intake and acknowledgment by the AI, formal report creation with reference number, routing to the manager via the agreed channel, 24-hour response commitment to the guest, manager escalation at 48 hours if no response. That's steps one through five in the to-be model, plus the escalation path. Step six — the personal manager response — goes to humans, as it should. Step seven — case closure confirmation — we can add in a second phase once the team is comfortable with the routing workflow.

Common Designer Mistake: Letting the Client Drive the Target to Near As-Is

The most corrosive mistake in the to-be selling conversation is allowing the client to negotiate the transformation target down to a point where it is essentially the current process with an AI interface layered on top. This happens gradually. The designer presents the to-be model. The client asks for modifications. The designer accommodates. The client asks for more modifications. The designer accommodates again. By the end of the conversation, the agreed workflow looks like the as-is, runs on the client's existing broken process, and carries the AI vendor's credibility.

This is not transformation. It is rebranding. And the designer who accepts it has become exactly what "A commodity AI provider automates the current process" describes.

The designer must know, before entering the room, what the minimum viable transformation target is — the lowest acceptable version of the to-be model that still meaningfully addresses the solution statement. This is not a negotiating position. It is a professional standard. Below this line, the designer is not building what they are claiming to build.

When the client drives the target toward the minimum line, the designer must name it: “I want to make sure we’re both clear about what we’re agreeing to. If we remove the manager response requirement, we have an AI that takes in complaints and creates records, but no committed path for those complaints to reach resolution. That means the guest’s experience ends with a confirmation that their complaint was logged — not that anything will be done about it. Is that the product you want to deliver to your guests?”

Most clients, faced with this question, find the organizational commitment. Some do not. The designer must know how to respond in both cases.

Design Practice: Selling the To-Be Model to a Founder

The following is a dialogue between an AI workflow designer and a founder who has developed a coaching and advisory service but cannot clearly communicate what it is or who it is for. The AI is being considered as a tool to help prospects understand and engage with the offer. The designer is attempting to sell the to-be model.

Designer: Before we talk about what the AI should do, I want to show you the to-be model we built from the simulation. This is what best-class resolution of your prospect’s situation looks like — before we know anything about what you currently have or don’t have. Let me walk you through it.

Founder: Sure.

Designer: A prospect comes in with a vague sense of a problem they can’t fully articulate. Maybe it’s a business not growing the way they expected. Maybe it’s a team that doesn’t execute. Maybe it’s their own clarity. The AI’s job in the to-be model is to let that problem emerge fully — not to diagnose it, not to pitch your offer, but to let the prospect hear themselves describe what they actually need. From that, the AI surfaces the solution statement: the condition that would make their problem no longer a problem. Then it creates a connection between their specific solution statement and a specific component of what you offer. The handoff to you is warm, specific, and contextualized.

Founder: I mean, can’t the AI just explain what I do and see if they’re interested?

Designer: It can. And that’s what most people build. The problem is that most of your prospects don’t know what they need until they’ve been heard. If the AI leads with your offer description, you get people who self-selected based on a description of your services — which means you get a lot of “this sounds interesting” conversations that don’t convert, because the prospect never connected their real problem to your offer. The to-be model creates that connection first.

Founder: But it would need to understand my offer to do that.

Designer: Yes. That's part of what we build. But before we build that, I need to ask you something directly: does your offer have enough internal structure that we can map specific components to specific problem types? Because if the answer to every problem type is "the founder works with you directly," we have an offer that is too undifferentiated to create a specific connection. The AI can only make a specific connection between a specific problem and a specific solution. If your solution is just "me," we have a clarity problem to solve before we have a workflow to design.

Founder: I don't think I need to change my offer. I need people to understand it better.

Designer: I hear that. But here's what the to-be model is showing us: the issue is not that prospects don't understand your offer. The issue is that they arrive without understanding their own problem clearly enough to know whether your offer addresses it. If we design an AI that explains your offer clearly to people who don't understand their own problem, we have improved the clarity of your pitch without improving the match rate. The prospect still doesn't know whether you can help them. They just know more about what you do.

Founder: So you're saying I need to change my offer.

Designer: I'm saying the to-be model requires that the offer be structured clearly enough that the AI can make a specific match. That may require clarifying the offer — not changing it fundamentally, but organizing it into specific problem types and corresponding value components. If you're not willing to do that work, the AI will be limited to pitching a general description. And that is a significant reduction in what the workflow can deliver.

Founder: How long would that clarification take?

Designer: One or two working sessions, if you come in with examples of your best client relationships — the ones where the fit was obvious and the work went well. We reverse-engineer the offer from those. The AI workflow follows from the clarity, not the other way around.

Key Takeaways

The to-be model sells movement from as-is toward to-be — not AI features. The designer's product is a direction and a design path, not a technology.

The gap presentation requires three frames: the to-be as best-class standard, the as-is as reality document, and the gap as transformation opportunity. All three frames must be established before the gap is shown.

The central question of the to-be selling conversation must be made explicit: do we bend the AI around the current process, or do we improve the process so the AI can deliver the to-be outcome?

Most of the transformation value is in process change, not AI capability. The AI follows whatever process exists. If the process is broken, the AI produces broken outcomes at scale.

Six categories of process change are most commonly required: defining ownership, creating records, changing staff behavior, establishing authority paths, defining status, and creating protocols.

Each form of client resistance has a specific honest response. The designer must prepare these responses before entering the room.

The transformation target is the minimum viable commitment — the lowest acceptable version of the to-be model that still meaningfully addresses the solution statement. Below that line, the designer is not building transformation. They are deploying rebranding. # Chapter 15: Designing for Authority, Status, and Trust

Opening Thesis

The design decisions that determine whether an AI workflow is trustworthy are almost never made in the prompt. They are made in the authority map and the status language. A workflow can be warm, well-phrased, empathetic in tone, and precisely calibrated to the customer’s emotional register — and still be fundamentally untrustworthy, because it makes claims it cannot support and transitions states that have not actually been reached.

Trust is not a voice characteristic. It is not a tone setting. Trust is the alignment between what the AI says is true and what is actually true at every moment in the workflow. When that alignment breaks — when the AI says “your refund has been issued” before the refund system has acted, or “we found your phone” before a staff member has confirmed the match — the customer’s trust in the institution is damaged in a way that no amount of warmth can repair. They were told something that was not true. They acted on it. They were wrong to act on it. That is the trust failure.

The authority map and the status language library are the design artifacts that prevent this failure. They are built before the conversational flow is written. The conversational flow is structured around them. This sequence is not optional — it is the defining discipline of trustworthy AI workflow design.

The Authority Map

The authority map is a design artifact that documents every step in the workflow where an external authority is required to make a state change real. It is the first design document produced after the to-be model is extracted from the simulation. It takes precedence over the conversational flow because the conversational flow must be shaped around it — not the other way around.

For each authority-bound step, the authority map documents five elements:

Who is authorized. The specific person, role, or system that has the authority to make this state change real. This is not the person who is usually around, or the person the AI is most likely to reach, or the person who would probably be okay with it. It is the person or entity whose action is required for the state to be legally, operationally, and organizationally real.

What they authorize. The specific state change that their action produces. Not a general description — a precise statement. “The manager approves the resolution offer” is imprecise. “The shift manager records a confirmed resolution action in the guest experience system, specifying the form and value of the resolution” is a precise authority definition.

How they signal authorization. The specific action, system event, or explicit signal that constitutes authorization. A verbal acknowledgment that is not recorded does not constitute authorization for the purposes of AI workflow design. The AI cannot detect an unrecorded acknowledgment. The signal must be detectable, auditable, and reliable.

What triggers the authorization request. The specific workflow event that initiates the authority request. The authority map defines this trigger precisely so that the AI can route to the authority actor at the correct moment — not too early (before the required information is assembled) and not too late (after the customer has been given a premature status signal).

Timeout and escalation. What happens if the authorized actor does not respond within the required timeframe. Every authority step must have a defined timeout and a defined escalation path. A workflow with undefined timeouts will produce false completion when authority actors are unavailable — because the AI has no instruction for what to do when the authority step is not completed.

The Status Language Library

Every authority-bound state transition in the workflow requires a corresponding honest status label. The status language library is the complete set of these labels, organized in sequence from initial intake to final resolution.

The following represents a standard library applicable to most service resolution workflows. The designer selects and customizes from this library based on the specific to-be model:

Drafted: The customer has described their situation and the AI has created an internal record. No external action has been taken. No commitment has been made.

Prepared: The AI has assembled all required information and has prepared the case for routing to an authority actor. The case is ready to submit but has not yet been submitted.

Submitted: The case has been routed to the appropriate authority actor. A routing event has occurred and is recorded. The authority actor has not yet acted.

Routed: The case has reached the designated authority actor’s attention queue. Delivery is confirmed. Action is not yet taken.

Pending Verification: The authority actor is actively reviewing the case. Verification is in process.

Verified: The relevant facts have been confirmed by the authorized actor or system. The verified facts now constitute the official record.

Approved: The authority actor has approved a specific action or resolution. The approval is recorded and may be executed.

Issued: A concrete artifact — a credit, a refund, a record update, a confirmed booking — has been created in the relevant system. Issuance is distinct from delivery.

Confirmed: The issued artifact has been delivered to or acknowledged by the recipient. Delivery is confirmed.

Completed: The resolution condition defined in the to-be model has been fully achieved. All authority steps have been executed. All status transitions have been made real. The case is closed.

Status is not bureaucracy. Status is the design layer that prevents false completion. Every status label is a truthful report of where the artifact or outcome actually stands. An AI that claims “Completed” when the status is “Submitted” has produced false completion — and the customer who received that claim may act on it, rely on it, and be harmed by it.

Forbidden Claims

For each authority-bound step, the designer documents a set of forbidden claims: specific language that must never appear in AI responses at or before that step in the workflow. The forbidden claims list is as important as the status library, because it provides the prompt engineer with the explicit boundaries within which the conversational AI must operate.

The following are examples of forbidden claims mapped to their associated authority conditions:

“Your refund has been issued” — forbidden until: the refund authority step has been completed and the issuance event has been confirmed in the billing system.

“We found your phone” — forbidden until: a staff member has physically confirmed a match between the found item and the customer’s description, and that match has been recorded.

“You are confirmed” — forbidden until: the reservation or booking system has been updated with a confirmed record, and the system has returned a confirmation event to the workflow.

“The complaint has been resolved” — forbidden until: the authority actor has taken a defined resolution action and the customer has been notified by a human contact with authority to speak to the resolution.

“The charge will be reversed” — forbidden until: the authority actor authorized to initiate a reversal has initiated it and the system has acknowledged the initiation.

“Someone will call you back” — forbidden unless: a callback is an actual committed protocol with a named responsible party and a defined timeline, and the authority to make that commitment has been confirmed.

The forbidden claims list converts the authority map into a production constraint. The conversational AI will, under some conditions, generate these claims naturally — because they represent the natural completion of the interaction from a conversational standpoint. The forbidden claims list overrides that natural completion. The AI must be prompted explicitly to hold these claims until the authority conditions are met.

Trust Architecture

The combination of the authority map, the status language library, and the forbidden claims list creates a trust architecture for the workflow. Trust architecture is the technical term for the full set of design decisions that align AI speech acts with actual states of the world.

A workflow with a complete trust architecture will never tell a customer that something is true when it is not yet true. It will tell the customer what is actually known, what is actually pending, and what is actually required before the next step can be confirmed. This may feel less satisfying in the moment of the interaction than a warmer, more definitive response. But a customer who is told “your report has been submitted and the manager will respond within 24 hours” is receiving an honest status signal. A customer who is told “this has been resolved” when it has not been resolved will call back angry, discover the misrepresentation, and lose trust that is very difficult to rebuild.

Warmth and accuracy are not in conflict. The AI can be warm and accurate simultaneously. The prompt should be designed to be accurate first and warm always. What the trust architecture prevents is accuracy being sacrificed for conversational satisfaction.

Designing the Authority Escalation Path

Every authority step requires an escalation path for the case where the authority actor does not respond within the defined timeout. The escalation path must be designed explicitly — not assumed, not left to informal convention.

The escalation path answers three questions:

Who is the backup? The secondary authority actor who can take the required action when the primary actor is unavailable. The backup must be genuinely authorized — not just adjacent in the org chart — or the escalation is fictional.

What does the AI tell the customer? While the authority step is pending and the timeout approaches, the AI must communicate an honest status. The communication must not be false completion. It must be accurate about the pending nature of the case and honest about the expected timeline. If the timeout is reached, the AI must communicate that the case is being escalated and provide a revised timeline.

What is the maximum hold time? The outer boundary after which the customer’s problem must be resolved through an alternative path — or the workflow must acknowledge that it cannot produce the outcome and provide an alternative. No workflow should hold a customer indefinitely with a “pending” status signal. Every pending state has a maximum duration. After that duration, an action is required.

The Production-Safe Prompt

The production-safe prompt encodes the authority architecture explicitly. It does not leave the AI to infer when authority steps are required or what status language is appropriate. It states these constraints directly.

A production-safe prompt segment for an authority-bound step looks like this:

At no point in this workflow should you indicate that a refund has been issued, approved, or confirmed unless the following condition has been explicitly flagged as true in the session context: [refund_authority_confirmed = true]. Until that condition is true, the most you may communicate regarding refund status is: “Your request has been submitted to the billing team for review. You will receive confirmation of the outcome within [X] business days.” Do not use the words “issued,” “approved,” “confirmed,” or “processed” in relation to the refund until the authority condition is met.

This level of specificity is necessary because conversational AI naturally gravitates toward resolution-language in service contexts. The training data for most language models includes enormous amounts of customer service interaction where agents confirm, resolve, and close cases — and the AI has learned to produce these closures as natural conversational endpoints. The production-safe prompt overrides this tendency explicitly.

Full Example: Authority Map and Status Library for the Lost-Phone Workflow

The following is the complete authority map and status language library for a lost-phone reporting and resolution workflow in a hospitality context.

Problem statement: A customer who dined at a restaurant believes they left their phone there and is contacting the restaurant to report it and attempt to recover it.

Solution statement: The customer either recovers their phone with confidence that the identification process was thorough and accurate, or receives honest confirmation that the phone was not found after a complete search, with a clear path for follow-up if it is found later.

Authority Map:

Step	Type	Authorized Actor	Authorization Action	Signal	Trigger	Timeout/Escalation
1. Item report created	Pattern	AI	Creates item report	Report ID generated	Customer describes item	N/A
2. Physical search request issued	Pattern	AI	Routes search request to staff	Search request logged	Report complete	N/A
3. Staff conducts physical search	Authority	On-duty floor manager or designee	Confirms search completed and records result	Search completion logged in system	Search request received	30 min; escalate to shift manager
4. Match assessment	Authority	Staff member who conducted search	Records “match found” or “no match” with description	Match status recorded	Search completed	N/A (immediate upon search)
5. Match confirmed	Authority	Floor manager	Confirms item identity against customer description	Confirmed match recorded	Match found status	15 min; escalate to general manager
6. Customer notified of confirmed match	Pattern	AI	Sends notification with pickup instructions	Notification sent log	Match confirmed	N/A
7. Item released to customer	Authority	Designated staff (not AI)	Customer identity verified; item released; release recorded	Release recorded in system	Customer arrives for pickup	N/A

Step	Type	Authorized Actor	Authorization Action	Signal	Trigger	Timeout/Escalation
8. Case closed	Pattern	AI	Case marked resolved	Closure event logged	Release recorded	N/A

Status Language Library (Lost-Phone Workflow):

Status	Meaning	Trigger
Reported	Customer has described item;	Item report ID generated
Search Requested	Staff search request issued	Search request logged
Search In Progress	Staff has acknowledged request; search underway	Staff confirmation received
No Match Found	Search completed; no matching item located	Staff records “no match”
Match Found	Staff has located an item matching description	Staff records “match found”
Match Confirmed	Floor manager has confirmed item identity	Manager confirmation recorded
Notification Sent	Customer has been notified of confirmed match	Notification log entry
Awaiting Pickup	Waiting for customer to claim item	Notification sent
Released	Item released to customer; identity verified	Release recorded
Completed	Case closed; item recovered	Release recorded
Closed — Not Found	Case closed; item not located after complete search	Staff final search confirmation

Forbidden Claims (Lost-Phone Workflow):

- “We found your phone” — forbidden until: Match Confirmed status achieved.
- “Your phone is here” — forbidden until: Match Confirmed status achieved.
- “We have it” — forbidden until: Match Confirmed status achieved.
- “Come pick it up” — forbidden until: Notification Sent status achieved.
- “The search is complete” — forbidden until: Search completion logged by authorized staff.
- “It wasn’t there” — forbidden until: “No Match Found” status recorded by authorized staff after complete search.

- “Your case is resolved” or “Completed” — forbidden until: Released or Closed — Not Found status achieved.
-

Common Designer Mistake: Authority Map as Afterthought

The most common and most damaging structural error in AI workflow design is treating the authority map as an afterthought — something to be documented after the conversational flow has already been built. The designer builds the conversation flow first, creates a satisfying user experience, writes warm and effective prompts, and then, in a later pass, attempts to retrofit the authority constraints.

This sequence produces authority maps that are incomplete, inconsistent with the conversational flow, and difficult to enforce without rewriting major sections of the prompt. More importantly, it produces a design philosophy where the conversational experience is primary and the trust architecture is secondary. That philosophy reliably produces false completion — because the conversational experience naturally gravitates toward satisfying completions, and the authority constraints that prevent false completions are treated as friction rather than as the core structural requirement they actually are.

Authority must be designed first. The authority map defines what can and cannot be claimed at each step of the workflow. The status language library defines the honest labels for each state. The forbidden claims list defines the boundaries of the production prompt. The conversational flow is then built within those constraints — and the result is a workflow that is simultaneously trustworthy and conversationally effective.

The reverse sequence — conversational flow first, authority constraints retrofitted — will always produce a design where some of the conversational choices conflict with the authority constraints, and the designer is left making compromises that reduce either the experience or the trust architecture. Neither compromise is acceptable.

Design Practice: Authority Map for the Double-Charge Workflow

Take the possible double-charge workflow for a mid-sized regional bank’s phone support process. Build the following:

Authority Map (minimum 6 authority steps):

Identify at least six steps in the resolution of a reported possible duplicate charge where an external authority is required to make a state change real. For each step, document: who is authorized, what they authorize, how they signal authorization, what triggers the request, and the timeout/escalation path.

Likely authority steps to consider: transaction lookup confirmation (system authority), duplicate determination (system or supervisor), provisional credit approval (supervisor or system threshold authority),

dispute initiation (customer explicit consent plus representative confirmation), formal investigation assignment (dispute processing team), resolution determination (dispute processing team or system), credit issuance confirmation (billing system), customer notification of final resolution (representative or system).

Status Language Library:

Write the full status progression for this workflow from “call received” to “case closed.” Provide an honest status label for each meaningful state change. There should be at minimum eight status labels.

Forbidden Claims List:

Write the forbidden claims list for this workflow. Consider carefully: what might the AI say naturally as a conversational completion at each of the authority-bound steps, and why would that claim be false completion? Write at least six forbidden claims, each with the authority condition that must be met before the claim can be made.

Production-Safe Prompt Segment:

Write one paragraph of the production-safe prompt that enforces the most critical authority boundary in this workflow. The most critical boundary is likely the boundary around credit issuance — the step where the customer is most likely to act on a false completion. The paragraph must explicitly name the authority condition, state what the AI is and is not allowed to claim before that condition is met, and provide the exact language the AI should use while the case is pending.

Key Takeaways

The authority map and the status language library are the primary design artifacts of trustworthy AI workflow design. They are built before the conversational flow. The conversational flow is structured around them.

The authority map documents, for each authority-bound step: who is authorized, what they authorize, how they signal authorization, what triggers the request, and the timeout/escalation path.

The status language library provides a truthful label for every meaningful state change in the workflow. Status is not bureaucracy — it is the design layer that prevents false completion.

The forbidden claims list converts the authority map into a production constraint. It specifies the exact language the AI is prohibited from using before each authority condition is met.

Trust is not tone. Trust is the alignment between what the AI says is true and what is actually true at every moment in the workflow. Warmth and accuracy must coexist; accuracy comes first.

The escalation path must be designed for every authority step. A pending state without a defined timeout and escalation path will produce false completion when authority actors are unavailable.

The production-safe prompt encodes the authority architecture explicitly. It does not leave the AI to infer when authority steps apply. Specificity is mandatory.

Designing the authority map as an afterthought produces a fundamentally compromised workflow. Authority must come first. # Chapter 16: Stewardship and Renewal Cadence

Opening Thesis

A production AI workflow is not finished when it launches. It enters a stewardship cycle. The steward's job is not to maintain the workflow in its original form — it is to keep the workflow pointed at the solution statement as everything around it changes. These are very different responsibilities, and the designer who treats them as equivalent will find that a well-maintained workflow gradually drifts away from the solution statement it was built to serve.

“Maintenance preserves the workflow. Stewardship renews the workflow around the solution.” Maintenance asks: is the workflow working the way it was designed? Stewardship asks: is the workflow still pointing at the right destination, and is there a better path to it than the one currently in production? Maintenance is essential but insufficient. It keeps the car running. Stewardship asks whether the car is still going where you need to go.

“Stewardship gives you renewal.” The renewal is not cosmetic. It is not a version bump or a prompt revision. It is the periodic practice of returning to the solution statement, asking whether the current workflow is still the best path to it, and being willing to redesign significantly if the answer is no. “Be willing to obsolete your own design.” This disposition — genuine willingness to replace a working design because a better design is now possible — is the steward's most important professional quality.

What Changes Over Time in an AI Workflow Environment

The first task of renewal stewardship is to understand what kinds of change affect an AI workflow and at what pace. An AI workflow is embedded in a dynamic environment, and the dynamics operate along several different axes simultaneously.

The AI Model Itself

AI models are updated, upgraded, replaced, and occasionally deprecated. New model versions may perform differently on the same prompts — better in some respects, different in others. A prompt that was carefully calibrated to a specific model's behavior may produce different output after a model upgrade, even if the prompt text is unchanged. New capabilities (improved tool use, better reasoning, expanded context windows, multimodal inputs) may enable workflow patterns that were not possible when the original design was built. Model deprecation may force migration to a new model under time pressure, with insufficient time for careful calibration.

The steward tracks model changes proactively. They do not wait for production incidents to discover that a model upgrade has changed the workflow's behavior.

Tools and Integrations

APIs change. Tools are versioned, and older versions are eventually deprecated. New tools become available that can handle steps the original design routed through workarounds. External systems that the workflow depends on change their data structures, authentication requirements, or reliability profiles. An integration that worked reliably at 100 calls per day may behave differently at 10,000 calls per day.

The steward maintains a live inventory of every integration in the workflow — including the version in use, the deprecation status of that version, and the next available version. Integration monitoring is a standard component of the stewardship plan.

The Business

Organizations change. Staff turns over. The person who was the named authority actor at a specific step in the workflow may leave the organization. A new manager may have a different authority delegation model. A process that was a clear line of responsibility may have become ambiguous after an organizational restructure. A restaurant chain that was operating regionally may have been acquired and must now conform to a national management structure that its original AI workflow does not support.

Volume changes. A workflow designed for 50 interactions per day may need to be reconsidered when volume reaches 500. Not because the AI cannot scale — in most architectures, the AI can — but because the authority steps designed for a 50-per-day volume may not be staffed or structured to handle 500. False completion risk rises nonlinearly with volume if the authority architecture has not been scaled with it.

Customer Behavior

How customers phrase things changes over time. Language evolves. New generations of customers bring different communication norms. Events in the world change what customers expect. A workflow designed for customers who call about charges may need to evolve when customers switch to messaging, or when an industry-wide incident changes what customers are worried about in a particular domain.

Vocabulary drift is a specific stewardship concern. A workflow that was calibrated to recognize certain phrasings as indicating certain problem types may produce incorrect categorizations when customer language has shifted. Periodic review of emergence patterns — returning to Method 1 and listening to recent conversations — is part of the renewal cadence.

The Solution Statement Itself

Perhaps the most important — and most often overlooked — category of change is the possibility that the solution statement has changed. The solution statement was defined at the beginning of the design process, based on emergence data collected at a specific moment. But what customers actually need, and what a resolved condition actually looks like, can evolve.

A restaurant complaint workflow built in one hospitality climate may face different customer expectations after a period of significant industry disruption. A billing dispute workflow built before a major regulatory change may have a solution statement that no longer reflects what the law requires the institution to deliver. The steward must periodically return to the solution statement and ask: is this still the right definition of resolution? If the world has changed, the solution statement may need to change with it — and if the solution statement changes, the entire workflow is subject to reconsideration.

The Renewal Cadence Design

A renewal cadence is a structured schedule of reviews designed to ensure the workflow is evaluated — not just monitored — at regular intervals and in response to specific triggering events.

Scheduled Renewal Reviews

The standard scheduled renewal interval for an active AI workflow is quarterly. This interval is short enough to catch model behavior drift and integration changes before they compound, and long enough to allow the collection of meaningful performance data between reviews.

For workflows in high-stakes domains — healthcare, financial services, legal contexts, or any domain where false completion has significant consequences — monthly reviews may be warranted during the first year of production operation.

Renewal reviews are also triggered by model upgrade events. Whenever the underlying AI model is updated — whether by the model provider or by the designer’s own prompt engineering — a formal renewal review is required. Model behavior can change significantly across versions, and a post-upgrade review is not a quality check on the new model. It is a recalibration of the workflow’s trust architecture to the model’s new behavior.

Incident Reviews

An incident review is triggered by specific production events rather than by a schedule. The incidents that require a formal review are:

- Any confirmed false completion event: the AI claimed a state was real that was not real.
- Any stuck-customer event: a customer was held in a pending state beyond the defined timeout without escalation functioning as designed.
- Any authority failure: an authority step was bypassed, timed out without escalation, or produced an incorrect state transition.
- Any volume anomaly: the workflow handled significantly more or fewer interactions than designed for, and the authority architecture showed strain.
- Any prompt change: whenever a prompt is modified in production, a formal incident-style review is warranted to confirm that the change did not introduce new false completion risks.

Incident Stewardship Protocol

The incident stewardship protocol is a structured response sequence that the steward follows when an incident is detected. It has six steps, and they must be followed in order.

1. Detect. The steward must have visibility into production events fast enough to detect incidents when they occur, not retrospectively. Detection requires monitoring infrastructure: completion rate tracking, abandonment rate tracking, false-completion incident flagging (often via human review spot-checks or customer complaint correlation), authority step response time monitoring, and escalation rate tracking. Detection is the steward's first design responsibility in a newly launched workflow — the monitoring infrastructure must be in place on launch day.

2. Classify. Once an incident is detected, the steward classifies its type before attempting to respond. The six incident types are: false completion event, tool failure, model drift (the model's behavior has changed without a prompt change), authority gap (the authority architecture failed to function as designed), prompt regression (a prompt change introduced a new failure mode), and external change (a business, staff, or system change that the workflow was not updated to reflect). Classification determines the response path. A model drift incident requires a different response than a tool failure.

3. Contain. Containment is the immediate action taken to prevent the incident from continuing while the root cause is diagnosed. Containment is not correction. A false completion incident may be contained by temporarily routing the affected interaction type to a human agent. A tool failure may be contained by disabling the tool call and routing to a fallback. Containment is always temporary and always explicitly acknowledged as such in the incident record.

4. Diagnose. The root cause analysis is conducted after containment, not during it. Diagnosis requires clean thinking, access to full interaction logs, and the discipline to identify the actual root cause rather than the most convenient narrative. The most common diagnosis error is attributing an incident to the AI's behavior when the actual root cause is an authority architecture gap or an organizational change that the workflow was not updated to reflect. The steward asks: where in the workflow did the failure originate? What was the first step where the outcome diverged from the design?

5. Correct. The correction is a thoughtful fix that addresses the root cause without introducing new problems. The steward resists the pressure to produce a quick patch — a prompt addition, a new exception branch, a one-off rule — without first considering whether the correction creates a new fragility elsewhere in the workflow. Every correction to an AI workflow prompt has the potential to change the model's behavior in situations the correction was not designed to address.

6. Document. After every incident, the steward updates the relevant design artifacts. If the incident revealed a gap in the authority map, the authority map is updated. If the incident revealed a missing forbidden claim, the forbidden claims list is updated. If the incident revealed a flaw in the status language, the status library is updated. The design artifacts are living documents, and every incident is a learning event.

Renewal Stewardship Protocol

The renewal protocol is the structured process for conducting scheduled renewal reviews. It is distinct from incident response: it is not triggered by a failure but by time and the possibility of improvement.

Step 1: Run the renewal audit. The renewal audit is a standard review format used at each scheduled interval. It covers: workflow performance metrics since the last review, any incidents since the last review and their resolutions, changes in the AI model since the last review, changes in tools and integrations, changes in the business that affect the workflow, changes in customer behavior patterns, and any changes in the domain or regulatory context.

Step 2: Ask the redesign question. This is the stewardship question that separates stewardship from monitoring: “If we designed this workflow today from scratch, with the current model, the current tools, and what we now know about how customers talk about this problem — would we design it the same way?” The honest answer to this question is almost always “not entirely.” The steward then asks which differences are significant enough to pursue.

Step 3: Identify available improvements. New model capabilities may enable patterns that were not possible at launch. New tools may eliminate steps that previously required workarounds. New customer behavior data may reveal that the workflow’s categorization logic is less effective than it was when calibrated on the original training transcripts. The steward catalogs the improvements now available and assesses the effort and risk of each.

Step 4: Re-run Method 2 if significant time has passed. If more than six months have passed since the original blind ideal simulation, or if the AI model has been significantly upgraded, the steward should re-run the simulation with the current model. The new simulation may produce a to-be model that is meaningfully different from the original — revealing new steps, new authority patterns, or a more direct path to the solution statement. Comparing the new simulation output to the current production workflow shows the renewal gap.

Step 5: Compare new to-be to current production. The gap between the new simulation’s to-be model and the current production workflow is the renewal opportunity. Some of that gap may represent aspirational improvements that are not yet achievable given current organizational constraints. Some may represent clearly achievable improvements that should be made in the current renewal cycle.

Step 6: Decide — renew, adjust, or keep. Based on the gap analysis, the steward makes one of three decisions. Renew: the workflow requires significant redesign to remain aligned with the solution statement and take advantage of current capabilities. Adjust: targeted improvements can be made without a full redesign. Keep: the current workflow is well-aligned and the available improvements are not material enough to justify the change risk in this cycle.

The Steward's Disposition

“Be willing to obsolete your own design.” This sentence is the steward's professional ethic. It is not easy to act on. A design that the steward has invested significant effort in building, calibrating, and improving carries psychological weight. The instinct is to preserve it, to patch it, to add to it — not to replace it.

The steward who is attached to the current design will find evidence for keeping it in every renewal review. They will classify available improvements as not material enough. They will interpret renewal gaps as acceptable trade-offs. They will gradually allow the workflow to drift from the solution statement while maintaining its operational integrity.

The steward who is genuinely willing to obsolete their design will find the renewal reviews clarifying. They will see the renewal gaps as information rather than as threats. They will make the redesign decision when the redesign decision is right — and they will trust that the quality of their design process is more valuable than the preservation of any particular design outcome.

Monitoring Design

Monitoring is the informational substrate of stewardship. Without good monitoring, the steward cannot detect incidents, cannot evaluate performance, and cannot produce the data that renewal reviews require. Monitoring design is therefore a stewardship responsibility, not an IT responsibility — the steward defines what needs to be visible, and the technical infrastructure is built to make it visible.

The key signals to monitor in a production AI workflow are:

Completion rate: The percentage of interactions that reach the resolution condition defined in the to-be model. Declining completion rate signals either a workflow problem or a shift in the problem population being handled.

Abandonment rate: The percentage of interactions where the customer exits before reaching resolution. Abandonment rate by stage reveals where in the workflow customers are dropping off — which may indicate confusion, frustration, or a status signal that is not meeting their expectations.

False-completion incidents: Confirmed events where the AI claimed a state that was not real. These should be tracked both in absolute count and as a rate. Any false-completion rate above zero warrants investigation.

Authority step response time: How long it takes the authority actor to complete their step after being triggered. This signal reveals organizational reliability issues that the workflow cannot resolve on its own but must route around.

Escalation rate: The percentage of interactions that trigger the escalation path. Rising escalation rate may indicate that the primary authority actors are overwhelmed, absent, or not responding — or that the trigger for escalation is set too low.

Customer satisfaction on resolution: Not on the conversation — on the actual resolution. A customer who received a warm, well-structured interaction but whose problem was not resolved is not a successful outcome. Measuring satisfaction on resolution requires follow-up survey design that specifically asks whether the problem was resolved, not whether the interaction was pleasant.

Full Example: One-Year Stewardship Cycle for the Bad-Restaurant-Experience Workflow

The following traces a one-year stewardship cycle for the workflow designed to handle post-dining complaint contacts at a regional restaurant chain.

Launch

The workflow launches with the authority architecture, status library, and forbidden claims list established in Chapter 15's terms applied to the complaint domain. The monitoring infrastructure is in place from day one: completion rate, abandonment rate, false-completion flag (triggered when a customer contacts the restaurant again about the same incident within 72 hours, indicating the first contact did not resolve the situation), manager response time, and escalation rate.

The launch team defines success thresholds: completion rate above 65%, manager response time under 24 hours for 85% of routed complaints, escalation rate under 10%, false-completion rate at zero.

90-Day Incident Review

At 90 days, monitoring data shows that manager response time is within target (22-hour median), but the escalation rate is 22% — more than double the target. Investigation reveals the cause: the workflow's 24-hour escalation trigger is set against clock time, not business hours. Complaints submitted at 11 PM on Friday are triggering escalation at 11 PM Saturday — correctly, technically, but the escalation is routing to the regional manager's personal contact channel during weekend off-hours, which the regional managers are ignoring.

Classification: authority architecture gap. The timeout design was correct in intent but incorrect in implementation — it did not account for business hour constraints on the authority actors.

Containment: the escalation trigger is temporarily paused for weekend night submissions, with an extended timeout until Monday morning.

Correction: the escalation logic is rebuilt to use business-hour-adjusted timeouts. A secondary authority path is designed for weekend evenings: complaints submitted after 9 PM Friday and before 9 AM Monday route to the weekend duty manager first, then to the regional manager if not acted upon by noon Monday.

Documentation: the authority map is updated with the business-hour-adjusted timeout logic and the weekend duty manager path.

6-Month Renewal Audit

At the six-month mark, a new version of the underlying AI model has been released, with significantly improved tool-use capability. The steward runs the renewal audit.

Performance metrics: completion rate 71% (above target), manager response time 18-hour median, escalation rate 8%, false-completion rate zero (one near-incident investigated and resolved). By metrics, the workflow is performing well.

The steward runs the redesign question: “If we designed this today from scratch with the new model?” The new model’s improved tool use means the complaint report creation step — which previously required the AI to prompt the customer for specific structured data across several exchanges — can now be done in a single natural-language summary with automated structured extraction. This would reduce the average intake step from 4.2 exchanges to 1-2 exchanges, a significant customer experience improvement.

The steward re-runs Method 2 with the new model. The simulation reveals that the new model also generates a more nuanced acknowledgment pattern — it can specifically reference the three typical failure categories (service, order, staff response) without being prompted, which was a gap in the original design.

Decision: adjust. The intake step is redesigned using the new tool capabilities. The acknowledgment pattern is updated to leverage the new model’s more nuanced output. The authority architecture is unchanged — the changes are to the pattern steps, not the authority steps, and do not require a full redesign.

12-Month Renewal Review

At twelve months, the restaurant chain has been acquired by a national hospitality group. The new management structure has three changes that directly affect the workflow: the “general manager” authority step now routes to a district manager in the national structure (the previous GMs are now “location managers” without authority to approve resolution actions); the personal response to the customer is now required to come through a branded guest experience team, not directly from the location; and a new CRM system is being rolled out across all locations.

The steward conducts a full renewal audit and concludes: renew. The organizational changes have altered two authority steps, the routing logic, and the system integration layer. The solution statement is unchanged — customers still need to feel genuinely heard and receive a meaningful acknowledgment with real consequences. But the path to that solution has changed structurally.

The renewal process begins with a new as-is discovery session (the acquisition has produced a new as-is), a comparison to the existing to-be model, and a fresh gap annotation. The authority map is rebuilt with the new authority actors. The CRM integration is redesigned for the new system. The personal response

step is redesigned for the branded guest experience team. The conversational flow is updated to reflect the new routing and timing parameters.

The renewal is substantial — it is not a version update. It is the workflow renewed around the solution statement, which has not changed, through a significantly changed organizational environment.

Common Designer Mistake: Confusing Monitoring with Stewardship

Monitoring tells you what happened. Stewardship asks what it means and what to do about it. The designer who has set up excellent monitoring and believes they are doing stewardship has made a category error — one that will eventually produce an outcome they cannot explain with their dashboards.

Dashboards show rates, volumes, and trends. They do not show whether the solution statement is still right. They do not show whether a better path to the solution is now available. They do not ask whether the authority architecture still reflects who actually makes decisions in the organization. They do not notice that a workaround has developed in the staff's handling of AI-routed complaints because the official authority path is too slow. They produce data. The steward produces judgment.

Monitoring is the input to stewardship. It is not a substitute for it. A steward who only monitors has outsourced judgment to a dashboard, and dashboards do not know what a solution statement is.

Design Practice: Stewardship Plan for the Founder-Offer Workflow

The following exercise develops a stewardship plan for a workflow in which a founder has deployed an AI voice agent to help prospects understand and engage with their offer.

Monitoring Design:

Identify the monitoring signals most relevant to this workflow. Consider: what does completion mean in this context? A prospect who completes the conversation but does not take a next step has not resolved the problem. A prospect who disconnects after two minutes may have found exactly what they needed or may have been immediately confused. What signal distinguishes these cases? At a minimum, define: the resolution condition (what does the workflow consider a successful outcome?), the completion rate metric (how is it calculated?), the abandonment-stage breakdown (where in the conversation are prospects most likely to exit?), and the follow-through rate (of prospects who reached the solution-statement moment in the conversation, what percentage took the defined next step?).

Incident Triggers:

Define the specific events that would trigger an incident review for this workflow. Consider: what would false completion look like here? (The AI claims the founder's offer is a match for the prospect's situation when it is not, and the prospect books a call expecting something the founder cannot deliver.) What would

a stuck-customer event look like? What would a model drift incident look like in this specific context — how would the steward know the model’s behavior had changed in a way that mattered?

Renewal Cadence:

Define the renewal schedule. Given that this workflow’s primary input is the founder’s offer, and offers evolve — particularly in early-stage businesses — the renewal cadence should account for offer changes as triggering events in addition to time-based reviews. Define: the standard review interval, the offer-change trigger, and the customer-behavior drift trigger.

Six-Month Renewal Audit Outline:

The founder has been operating for six months. The offer has evolved: what began as a general business advisory service has narrowed to a specific methodology for teams in a specific industry. The AI workflow was calibrated to the original broader offer. Write what the six-month renewal audit would examine, what the redesign question would surface, and what decision the steward would most likely make. Consider: has the solution statement changed? Has the problem-to-offer connection changed? Would a re-run of Method 2 with the current offer produce a meaningfully different to-be model?

Key Takeaways

A production AI workflow enters a stewardship cycle at launch. The steward’s job is not to preserve the workflow in its original form but to keep it pointed at the solution statement as everything around it changes.

Five categories of change affect an AI workflow over time: the AI model, the tools and integrations, the business, customer behavior, and the solution statement itself. The steward tracks all five.

The renewal cadence consists of two types of review: scheduled reviews (quarterly, or triggered by model upgrades) and incident reviews (triggered by specific production events).

The incident stewardship protocol has six steps that must be followed in order: detect, classify, contain, diagnose, correct, document. Containment is not correction. Documentation closes the loop and updates the design artifacts.

The renewal stewardship protocol asks whether the current workflow is still the best path to the solution statement. The redesign question — “would we design it the same way today?” — is the central question of every renewal review.

Monitoring is the informational substrate of stewardship, not its equivalent. Monitoring produces data. Stewardship produces judgment. A designer who only monitors has outsourced judgment to a dashboard.

The steward’s most important professional quality is the willingness to obsolete their own design. Attachment to the current design is the primary failure mode of long-term stewardship. # Part Four: Case Studies

Reading These Cases

The chapters in Part Four apply the full ten-phase life cycle to six discrete scenarios. Each case begins with a messy human expression — unfinished, emotionally real, and difficult to act on — and works forward through problem statement, solution statement, blind ideal simulation, pure to-be model, as-is discovery, gap analysis, selling conversation, authority mapping, and stewardship. Each case ends with the question that every workflow must eventually answer: what are we watching, and what will we do when something drifts?

The scenarios chosen here are deliberately simple. A lost phone. A possible double charge. A bad dinner. A sales call. A founder who cannot yet explain what they are selling. Simple scenarios are not trivial scenarios. They are the ones where the method is most legible, because complexity of subject matter does not obscure the structure of the reasoning. If you can trace the full arc from messy human expression to a renewed workflow in a lost-phone case, you can trace the same arc inside a hospital discharge protocol or a multi-party contract approval chain. The method does not change. The stakes change. The cast of actors changes. The pattern-recognition burden on the AI participant changes. The method does not.

A common mistake in workflow design education is to open with elaborate scenarios — enterprise integrations, multi-department handoffs, regulatory compliance chains — on the assumption that complexity demonstrates seriousness. This is an error. Complexity obscures the joints. When you work through a simple case with full rigor, every joint is visible. You can see exactly where pattern recognition ends and authority begins. You can see exactly what false completion looks like. You can see the difference between a workflow that resolves and a workflow that merely moves.

Read these cases as architecture, not as templates. No real scenario will match them exactly. What transfers is not the specifics — not the fourteen-day lost-item policy, not the provisional credit threshold — but the shape of the thinking. Problem statement naming the real gap. Solution statement naming the real resolved condition. The AI as a pattern-bearing participant who can hold, structure, and route — but cannot decide, authorize, or promise.

Each case also demonstrates something harder to name but essential to practice: the discipline of not resolving too early. When a caller is distressed about a lost phone, the temptation is to say “we’ll look into it” and move on. That is not resolution. Resolution is a found phone, or a documented loss with a clear policy and a definite owner. The cases model what it looks like to hold that standard without losing the human texture of the exchange.

Work through each case in full before moving to the next. The method compounds. By Chapter 21, you will be applying the same ten phases to an interior problem — the problem of an idea not yet expressible — and the accumulated fluency from the earlier cases will carry you through terrain that would otherwise feel formless. # Chapter 17: Case Study — Lost Phone at a Restaurant

Overview

This case begins with one of the most ordinary distress calls a business receives: someone who left something behind and is not certain what they left, when they left it, or whether it is still there. The scenario is common enough that most restaurants have some informal version of a response — someone checks the lost-and-found, someone looks under the table — but informal versions of a response are not workflows. They have no documentation, no ownership, no policy, and no defined resolution condition. They have only the hope that someone will remember to look.

The ten phases worked through here will show how even the simplest scenario carries enough complexity to warrant a structured workflow, and how the AI participant fits precisely into that structure without overreaching it.

1. Messy Human Expression

The following is a near-transcript of a caller reporting a lost phone. It is reconstructed from the kinds of calls restaurants regularly receive, and reflects the emotional and cognitive state of someone who has realized something is missing.

Phone rings. Staff member answers.

Caller: Hi, um — I was in last night, I think — I mean, I was definitely in last night, I just — I'm not sure exactly what time. It was maybe seven? Seven-thirty? We had a reservation, it was under, um, McKinley — no, wait, that's my maiden name. It might be under Harmon. It was two of us, at a table — I think it was kind of toward the back, near the window, but I'm not positive because we asked to be moved. Anyway, I think I left my phone. I've been — I've been looking everywhere this morning and I just, I can't find it, and I've checked my car, I checked my coat pockets, I — yeah.

Staff: Oh, okay. Let me —

Caller: It's a black iPhone. Thirteen, I think? Or fourteen, I don't — I bought it a year ago. It's got a case, a dark blue case, kind of a navy blue, with — I think there's a little crack on the bottom left corner of the case, not the screen, just the case. The screen's fine. I mean — I think the screen's fine. I'm really hoping.

Staff: Okay, I can check the —

Caller: And the thing is, I have photos on there. My daughter's recital was last week and I haven't backed them up yet, I know, I know, I should back them up, but I haven't. And all my contacts. I don't — I mean I don't know anyone's phone number anymore. I don't have my mom's number memorized. It's just, it's a lot. I'm sorry, I know this is — this is probably a normal thing for you, I'm just kind of freaking out a little.

Staff: No, it's okay, I understand. Do you remember where —

Caller: The thing is, I might have left it in the booth? We were in a booth, actually, not a table — we switched from a table to a booth. And I remember putting it on the seat next to me, and then we were leaving and it was kind of rushed because my husband — anyway, we left quickly. And I don't know if someone handed it in or what. I called it this morning but I can't tell if it rang because my ringer might have been off. I also tried to find it on Find My iPhone but it says the location is — I don't know, it says last seen at, it shows near your address, which is good, I think? So maybe it's there. But I'm also wondering — what if someone picked it up? What if it's not there anymore? Do you have like a lost and found? Is there a box or something? Also, do you — sorry — do you have a policy for how long you keep things? Because I can come get it today, I can definitely come today, I'm free this afternoon.

This transcript is long, recursive, and emotionally weighted. It contains at least seven distinct information threads: reservation name uncertainty, table location uncertainty, device description, case description, content anxiety (photos, contacts), location signal from Find My iPhone, and a question about retrieval policy. It also contains an emotional disclosure — the recital photos — that is not operationally relevant to finding the phone but is highly relevant to the quality of the response.

2. Problem Statement

The caller's phone is missing. She believes she left it at the restaurant the previous evening, and she has reasonable evidence (Find My iPhone location signal) suggesting it may still be on the premises. She does not know whether the phone has been found and logged, whether it has been moved, or whether a retrieval process exists.

The full problem statement: a customer has lost a personal device that contains irreplaceable data. She has partial location information suggesting the device may be recoverable. No confirmed retrieval process exists from her vantage point. She cannot determine whether the phone is safe, who is responsible for it, or how to retrieve it. The gap between her current state and recovery is: documentation of the loss, physical search of the reported location, a responsible party named, and a defined retrieval path.

3. Solution Statement

The solution statement is: found phone.

More precisely: the phone has been physically located, the customer has been notified, and a confirmed retrieval or return has been completed. If the phone is not found after a thorough search, the solution statement is: documented loss with a clear policy communicated to the customer and a named staff contact who will follow up if the phone surfaces.

The solution statement is not “we checked.” Checking without documentation is not resolution. Checking without a named responsible party is not resolution. A warm phone call response that ends with “I’ll let someone know” is not resolution. The solution is either the phone in the customer’s hand or a documented, owned process with a defined endpoint.

4. Blind Ideal Simulation Summary

When a well-configured AI participant receives this call in a demonstration environment — without any existing workflow, policy, or staff process in place — it will typically generate an idealized response sequence that looks approximately like this:

The AI greets the caller and immediately signals that it will help and take information seriously. It asks for the caller’s name, the reservation date and approximate time, a description of the item, and the last-known location within the restaurant. It structures the information into a formatted intake record as the conversation progresses. It acknowledges the emotional content — the photos, the anxiety — briefly and without dwelling, returning quickly to the information-gathering task. It informs the caller that the restaurant has a lost-item policy, that a designated staff member will conduct a physical search within a defined timeframe, and that the caller will receive a follow-up contact within a specific window (e.g., within two hours). It offers to send the caller a confirmation message with a reference number and the name of the staff contact.

This simulation is useful because it exposes what the workflow requires. It requires: an intake form or log, a search protocol, a designated staff owner, a follow-up timeline, a contact method, and a policy on how long items are held. Without these, the AI is generating the shape of a resolved workflow over the absence of one.

The simulation is not the workflow. It is the signal that points toward what the workflow must contain.

5. Pure To-Be Model

The following is the structured to-be workflow for lost-item handling, derived from the ideal simulation and refined against operational reality.

Step 1 — Intake The AI or staff member receives the report. A structured intake record is created immediately, capturing: caller name, contact number, date and time of visit, table or booth location, item description (type, color, distinguishing features), and any supplementary information (e.g., Find My iPhone signal).

Authority checkpoint: AI may create and populate the intake record. AI does not conduct the search or make any claim about whether the item has been found.

Step 2 — Notification to Designated Staff Owner The completed intake record is routed immediately to the designated lost-item staff owner — a specific role, not “whoever is available.” The owner receives a notification through the restaurant’s established channel (text, app, or manager dashboard).

Authority checkpoint: The designated owner is a human role defined in advance. The AI routes to that role; it does not search, decide, or respond to the customer beyond intake confirmation.

Step 3 — Physical Search The staff owner conducts a physical search of the reported location within a defined timeframe (recommended: within 30 minutes of intake during operating hours, or at opening the following morning if reported after close). The search covers the reported location and secondary locations (host stand, lost-and-found box, manager’s office).

Authority checkpoint: Physical search is a human action. Findings are logged in the intake record.

Step 4 — Customer Contact If the item is found: the staff owner contacts the customer at the number provided, confirms the item is secure, describes the item to verify identity, and arranges retrieval. If the item is not found: the staff owner contacts the customer, communicates that a search has been completed, describes what was checked, and confirms that the item will be flagged in the log for 14 days.

Authority checkpoint: Retrieval arrangements are made by the staff owner. The AI does not make promises about retrieval, timing, or item condition.

Step 5 — Log Maintenance The intake record remains open and visible in the lost-item log for 14 days from the date of the report. If the item is found during those 14 days, the record is updated and the customer is notified immediately. At 14 days with no recovery, the record is closed and the item (if found after the fact) may be donated or disposed of per policy.

Authority checkpoint: Log closure and item disposition decisions are made by the designated owner or manager.

6. As-Is Discovery Questions

Before designing the to-be workflow, the designer must understand how lost items are actually handled today. The following questions constitute the as-is discovery interview.

1. When a customer calls to report a lost item, who answers the phone? Is it always the same role, or does it vary by time of day or shift?
2. What happens in the first sixty seconds of that call? Walk me through exactly what you do.
3. Is any information written down during that call? Where — on paper, in a system, on a sticky note?
4. After you take the information, who does it go to? How does it get to them?
5. Is there a designated person responsible for lost items, or does it depend on who is around?

6. How long after a call does a physical search typically happen? What does “physical search” mean in practice — where do you check?
 7. Is there a dedicated lost-and-found location in the restaurant? Who has access to it?
 8. When you find an item, how do you contact the customer? Do you have their number from the call, or do you have to find it another way?
 9. If the item is not found, do you follow up with the customer? How, and when?
 10. Is there a policy for how long you hold found items before disposing of them? Is that policy written down anywhere?
 11. Has there ever been a situation where a customer came to pick up an item and it had been misplaced, given to someone else, or lost again internally? What happened?
 12. When do you escalate a lost-item situation to a manager? What would make you do that?
 13. Are staff trained on how to handle these calls, or is it informal knowledge passed between employees?
 14. Is there any record you can pull up right now that shows all the lost-item reports from the past month?
-

7. Likely As-Is Gaps

Based on typical restaurant operations, the following gaps will be present in most as-is environments.

No structured intake. Most restaurants have no intake form for lost items. Information is taken verbally, possibly jotted on paper, and often not retained beyond the shift.

No designated owner. “Whoever is available” is not an owner. Responsibility diffuses. When everyone is responsible, no one is responsible.

No defined search timeline. “We’ll check” does not specify when. A search that happens at the end of a busy dinner service — six hours after the report — is not the same as a search that happens within thirty minutes.

No follow-up protocol. If the item is not found on the first check, most operations have no process for subsequent checks or for contacting the customer with a negative result.

No retention policy. Items accumulate in drawers, on shelves, and in manager offices with no defined disposition timeline. Staff cannot tell a customer how long the item will be held because no one has decided.

No log. Without a log, there is no institutional memory. A phone lost on a Tuesday that is found on Thursday will not be connected to the Tuesday caller unless someone happens to remember.

No training. Staff handling these calls have not been told what information to collect, in what order, or what to promise.

8. To-Be Selling Conversation

The following dialogue excerpt demonstrates how a workflow designer presents the to-be model to a restaurant manager. The designer is proposing three concrete changes: a physical lost-item log, a designated staff owner, and a 14-day retention policy.

Designer: Right now, when someone calls about a lost item, the information lives in whoever answered the phone's memory. Does that match what you see?

Manager: Yeah, pretty much. Sometimes they write it down. Sometimes they tell me.

Designer: And if the person who took the call isn't in the next day, what happens to that information?

Manager: Honestly? It's probably gone.

Designer: So what we're designing is a process where the information can't disappear, regardless of who takes the call. Three things need to exist that don't currently exist. First: a log. A physical book or a shared digital form — doesn't matter which — where every lost-item report gets written down with the same fields every time. Caller name, number, date of visit, location in the restaurant, item description. That log lives in one place. It's always there.

Manager: Okay, that makes sense.

Designer: Second: one person owns it. Not "whoever is free." One designated role per shift who is responsible for conducting the search within thirty minutes of a report, updating the log, and making the call back to the customer. That person knows they own it before the shift starts.

Manager: What if that person gets busy?

Designer: Then they hand it off explicitly to another named person. That handoff gets noted in the log. The point is: at every moment, one person's name is attached to every open report.

Manager: And the third thing?

Designer: A policy. Fourteen days. If an item is found and logged, you hold it for fourteen days from the date it was reported. After that, you can donate it, dispose of it, or do whatever makes sense for your operation. But the customer gets told that policy upfront, when they call. Not "we'll keep it as long as we can." Fourteen days. That is a promise you can keep.

Manager: What if someone comes in on day fifteen?

Designer: Then you tell them the policy and you apologize. That is a better outcome than what happens now, which is someone comes in to pick up a phone that no one can find because it was never logged.

9. Authority Boundaries

The authority map for this workflow divides cleanly between pattern recognition and decision authority.

AI participant — pattern layer: - Receive and structure the intake call - Populate the intake form with caller-provided information - Acknowledge emotional content appropriately without dwelling - Confirm to the caller that the report has been received and logged - Route the completed record to the designated staff owner - Communicate the restaurant’s stated policy (retention period, search timeline) if that policy exists and has been provided to the AI

Human staff — authority layer: - Conduct the physical search - Determine whether a found item matches the description - Contact the customer with results - Arrange retrieval - Make any exception to the standard 14-day policy - Decide what to do with unclaimed items after retention period - Escalate to manager (e.g., if a found item shows signs of damage, if the customer claims mishandling, if a second customer claims the same item)

Manager — elevated authority layer: - Approve any exception to retention policy - Handle disputes about item condition - Decide disposition of items after retention window

The critical boundary is item identification. The AI may not tell a caller that their phone has been found. It cannot compare a caller description against a found item and declare a match. That determination requires a human eye, physical access to the item, and the authority to make a claim on behalf of the restaurant. A premature match — “yes, we have a black iPhone” followed by “actually that was someone else’s phone” — is not a minor error. It is a trust failure with downstream consequences.

10. Stewardship Risks

A well-designed workflow does not remain well-designed indefinitely. The following risks require monitoring over time.

Risk: AI matches a description prematurely and causes false completion. If the AI is given access to the lost-and-found log and is asked to cross-reference incoming reports against found items, it will make pattern matches based on description. Those matches will sometimes be wrong. A caller who is told “it sounds like we may have your phone” and then arrives to find the phone belongs to someone else has been harmed by a false positive. Monitor for any workflow configuration that routes AI pattern-matches directly to the caller without human verification as the intervening step.

Risk: Staff stop updating the log. The log is only as useful as its entries. In high-volume or high-stress periods — busy weekend services, staff turnover — log updates slip. When a new report comes in and matches an unlogged found item, the match cannot be made because the found item was never recorded. Audit the log monthly. The stewardship question is not “does the log exist” but “is the log current.”

Risk: Manager notification goes unread. Routing a notification to a manager is not the same as a manager acting on it. If the designated staff owner is unavailable and the escalation path runs to a manager who receives high volumes of messages, the lost-item notification may be read but not acted on within the required window. Stewardship requires periodic checks: are notifications being read? Are they being acted on within the defined timeframe? If the answer to either is no, the escalation path needs redesign, not a reminder.

Risk: Policy drift. A 14-day policy communicated verbally to staff will drift. Some staff will say seven days. Some will say “as long as we can.” Some will say whatever they think the caller wants to hear. The policy must be written, visible, and part of onboarding for new staff. Review the policy annually and whenever there is significant staff turnover.

Risk: Workflow orphaned after staff change. The designated owner role is often a specific person rather than a defined role with documented training. When that person leaves, the workflow may effectively cease to exist. Stewardship requires that the role definition — not just the person — is documented and transferred. The workflow should survive the departure of any single individual. # Chapter 18: Case Study — Possible Double Charge

Overview

A customer who believes they have been charged twice for the same transaction is not primarily in a billing dispute. They are in a state of financial anxiety. Whether the duplicate charge is real, pending, or a processing artifact does not change the initial emotional register of the call. The workflow designer must account for both the technical question — is there a duplicate charge, and if so, what is the correct remedy — and the human question, which the customer is asking even when they are not using these words: am I being taken advantage of, and can I trust you to fix it?

This case is instructive for workflow design because it sits precisely at a boundary that many organizations handle poorly. The AI participant can do significant work in this scenario — explaining the difference between pending and posted transactions, acknowledging the caller’s concern, and gathering the information needed to investigate. But the AI participant cannot issue a provisional credit, cannot authorize a refund, and cannot represent that the charge is definitively an error. These are authority actions, and this case maps that boundary in detail.

1. Messy Human Expression

The following is a near-transcript of a customer calling a business to report what appears to be a double charge.

Phone connects.

Caller: Hi, um, yes — I'm calling because I think you charged me twice for the same thing? I'm looking at my banking app right now and I'm seeing — okay, so there's one charge from you from — it says the fourteenth, and then there's another one that says — hold on, let me scroll — it says pending, but it's also from you, and it's the same amount. Or it looks like the same amount. It's thirty-seven dollars and — let me look — thirty-seven-eighteen and then thirty-seven-eighteen, so they're exactly the same. And I'm like — is that normal? Is that supposed to happen?

Agent: Okay, let me take a look —

Caller: Because here's the thing, I need to be careful right now, I have rent due on the first and I've been watching my account really carefully. And I'm not trying to cause a problem, I'm not that kind of person, I just — if you took the money twice, I need to know. Can you see my account?

Agent: I can pull up your —

Caller: Because I've been checking the app all morning and the pending one is still there. It hasn't — it hasn't gone away. I thought pending ones go away? Or do they post? I don't know how that works exactly. Like sometimes I see pending and then it just disappears, and sometimes pending becomes posted, and I'm not sure which one is which on my bank. I probably should know this. I just don't. I'm sorry.

Agent: No, that's a fair question —

Caller: And also — I don't want to be dramatic, but I noticed this because I got a notification from my bank, and so I went in and I looked, and I thought maybe it was just a glitch in the app. But it's been there since yesterday. And now I'm also wondering — is it possible that maybe I authorized it twice by accident? Like, maybe I clicked something twice? I do that sometimes. But even if I did, like, you shouldn't charge me twice for the same thing, right? That's not how it's supposed to work.

Agent: You're right, that shouldn't happen —

Caller: Okay so here's what I want to know. I want to know: one, is that second charge real, two, if it's real was it a mistake, and three, what are you going to do about it. Because I don't have — like, if I have an extra thirty-seven dollars sitting somewhere that belongs to me, I want it back. Especially right now.

This transcript is structurally cleaner than the lost-phone call — the caller is anxious but organized, and she has in fact identified the three questions she needs answered. What makes it difficult operationally is precisely what she has identified: the distinction between pending and posted transactions, which is a technical concept that most customers do not fully understand and which many front-line staff do not explain clearly or consistently. The caller's confusion about this distinction is not ignorance; it reflects the genuinely ambiguous way banking apps display transaction states. The workflow must address this confusion as a structural feature, not as a one-off correction.

2. Problem Statement

The customer has observed two line items on her banking app, identical in amount, both attributed to the same vendor, one marked posted and one marked pending. She does not know whether this represents a genuine double charge, a duplicate authorization hold, a processing artifact, or an error she may have introduced herself. She has rent due and is tracking her account closely. She cannot determine from the app alone whether she has been overcharged.

The full problem statement: the customer has lost financial security because she cannot determine whether her account has been charged twice for a single transaction. She lacks the vocabulary and access required to evaluate the situation herself. The gap between her current state and clarity is: a technically accurate explanation of what she is seeing, an investigation of whether a duplicate charge occurred, and if so, a corrective action with defined authorization and timeline.

3. Solution Statement

The solution statement is: financial security restored.

More precisely: the customer understands what she is seeing in her banking app, knows definitively whether she was charged twice, and if she was, has received either a confirmed reversal or a provisional credit with a defined investigation timeline. Her account balance accurately reflects only the charges she authorized.

The solution statement is not “we explained it to her.” Explanation without investigation is not resolution when the investigation is warranted. The solution is not “we opened a ticket.” An open ticket is a process in motion, not a resolved condition. Financial security restored means the customer ends the interaction knowing the outcome — or knowing exactly when and how she will know the outcome.

4. Blind Ideal Simulation Summary

In a demonstration environment, a well-configured AI participant handling this call will generate an idealized response sequence roughly as follows:

The AI greets the caller and quickly acknowledges the concern. It asks for account or transaction information to pull up the record. It explains — clearly and without condescension — the difference between pending and posted transactions: a pending transaction is an authorization hold, not yet settled; it may convert to a posted charge, or it may fall off without posting, depending on whether the merchant settles the payment. It acknowledges that seeing two identical amounts, one pending and one posted, can indicate either a processing artifact or a genuine duplicate charge, and that only a review of the transaction records can determine which.

The AI then gathers the specific transaction details (dates, amounts, authorization codes if available) and logs them. It informs the caller that the investigation will be conducted within a stated timeframe (e.g., three to five business days), that she will be contacted at her preferred number or email, and — if the organization has a provisional credit policy — that a provisional credit may be issued in the meantime, pending supervisor authorization.

The AI confirms the caller’s contact information and provides a reference number. It does not promise a refund. It does not tell her the charge is definitely a mistake. It does not issue the provisional credit itself.

The simulation reveals what the workflow requires: a clear explanation script for pending versus posted transactions, an investigation protocol, a provisional credit authorization threshold and process, and a defined customer callback protocol.

5. Pure To-Be Model

Step 1 — Intake and acknowledgment The AI or staff member receives the call. The caller is acknowledged promptly. A structured intake record is created: caller name, account or order number, transaction date, transaction amount, description of what is seen on the banking app (posted vs. pending, exact amounts), and what the customer wants as an outcome.

Authority checkpoint: AI creates the intake record. AI does not access account records directly unless it has been explicitly granted that access within the organization’s security architecture.

Step 2 — Transaction education The AI provides a clear explanation of the difference between pending and posted transactions. This explanation is not improvisational — it is a consistent, approved script that the organization has verified for accuracy. The explanation covers: what pending means, when it resolves, under what conditions it converts to a posted charge, and under what conditions it falls off without posting.

Authority checkpoint: Education is a pattern action. The AI may provide the approved explanation. The AI does not diagnose the specific transaction; it provides the framework the customer needs to understand what she is seeing.

Step 3 — Information gathering for investigation The AI or staff member gathers the specific transaction details: date(s), amount(s), any reference or confirmation numbers the customer has, the customer’s bank and whether she has a transaction ID from the banking app. This information is logged in the intake record and passed to the investigation queue.

Authority checkpoint: Information gathering is a pattern action. AI routes the completed record to the billing investigation role.

Step 4 — Provisional credit assessment A billing staff member or supervisor reviews the intake record. If the transaction record confirms a duplicate charge, the billing staff member proceeds with the reversal.

If the situation is ambiguous and the customer meets the threshold criteria (e.g., charge amount is above a defined floor, customer has rent or stated hardship), the supervisor may authorize a provisional credit while investigation continues.

Authority checkpoint: Provisional credit issuance requires supervisor authorization. This is a human authority action and is not delegated to the AI.

Step 5 — Customer callback The billing staff member or supervisor contacts the customer within the defined investigation window (recommended: within 24 hours for amounts above a threshold; within three business days otherwise). The callback provides: determination of whether the charge is a duplicate, the action being taken, and the timeline for any credit to appear on the account.

Authority checkpoint: Outcome communication is a human action when it involves a financial determination. The AI may send an interim confirmation message (“your concern has been logged and will be reviewed within X hours”) but does not communicate the final determination.

6. As-Is Discovery Questions

1. When a customer calls about a possible duplicate charge, who takes the call? Is that role trained specifically on billing questions, or is it general customer service?
2. Can the person taking the call see the customer’s transaction history in real time? What system do they use?
3. Is there a standard explanation staff give for the difference between pending and posted transactions? Is it written down anywhere, or is it informal?
4. Walk me through what happens if the customer has both a pending and a posted charge for the same amount. What is the first thing you tell them?
5. At what point does a front-line staff member escalate to a supervisor or billing specialist? What triggers that escalation?
6. Does the organization have a provisional credit policy? If so, what is the threshold for issuing one, and who can authorize it?
7. How long does a billing investigation typically take from intake to resolution?
8. How does the customer find out the result of the investigation? Who calls them, and by when?
9. Is there a callback protocol for billing investigations? Is it documented?
10. Has there ever been a case where a customer was told the charge was not a duplicate and it turned out it was? What happened?
11. Are there any cases where a pending charge converted to a post and the customer was charged twice? How was that handled?

12. Is there a process for documenting the outcome of billing investigations? Where does that documentation live?
 13. If a front-line staff member tells the customer “we’ll look into it,” what does “look into it” mean in practice — who does it, how, and by when?
 14. What does a staff member do when a customer says they need the money back today because of rent or an urgent expense? Is there any flexibility in the process for those situations?
-

7. Likely As-Is Gaps

No standard script for explaining pending versus posted transactions. Front-line staff handle this question differently depending on their own understanding and experience. Some give accurate explanations. Some give inaccurate ones (e.g., “pending means it hasn’t hit yet, so you’re fine” — which is not always true). Inconsistency in this explanation is a direct source of customer confusion and repeat calls.

No clear authority threshold for provisional credit. Many organizations have an informal understanding that “supervisors can issue credits,” but no defined threshold for when they should, what documentation is required, and within what timeframe. This produces inconsistent outcomes: one customer in an identical situation receives a provisional credit and another does not, based on which supervisor was on duty and how that supervisor felt about the call.

No customer callback protocol after investigation. Investigations are initiated but outcomes are often not communicated proactively. Customers are told to “check back in a few days.” When the investigation is complete, no one calls the customer unless the customer calls again. The customer who is tracking her rent balance cannot monitor an investigation she has no visibility into.

Confusion between company records and bank records. Front-line staff can see the company’s transaction records, which show what was authorized and what was settled. They cannot see the customer’s bank app, which shows pending holds, authorization codes, and timing differently. Many escalation calls occur because neither party is looking at the same data, and no one on the front-line team has been trained to bridge that gap.

No defined investigation queue. “We’ll look into it” often means “I’ll tell my supervisor.” Whether the supervisor looks into it that day or three days later depends on the supervisor’s workload and whether the inquiry reappears. There is no formal investigation queue with defined handling times.

8. To-Be Selling Conversation

The following dialogue excerpt shows the designer presenting the to-be model to a billing operations manager.

Designer: When a customer calls about a possible double charge, what's the most common thing they're confused about?

Manager: Whether it's actually a double charge or just a pending thing that will go away.

Designer: Right. And is there a script for that?

Manager: Not really. People kind of explain it in their own words.

Designer: So some people explain it correctly and some people explain it in a way that might not be accurate?

Manager: Probably, yeah.

Designer: That's the first thing we need to fix. One approved explanation, written down, that every staff member uses — including the AI if it's handling that part of the call. It's not a long script. It's three sentences. But it has to be accurate, and it has to be consistent, because the customer who calls back in two days and hears a different explanation from a different staff member is going to distrust the whole operation.

Manager: That's fair.

Designer: The second thing is provisional credits. Right now, who can issue one?

Manager: A supervisor. But it's kind of informal — it depends on the situation.

Designer: What we need is a threshold. Not a rigid formula, but a decision rule. Something like: "If the charge is over fifty dollars and the customer has stated financial hardship, a supervisor can authorize a provisional credit within four business hours of intake." That's not a promise to every customer. It's a defined process that your staff can rely on and that you can audit. Right now, you can't audit it, because it's not documented.

Manager: What if we set the threshold and it gets abused?

Designer: You set the threshold, you document every issuance, and you review it monthly. If it's being abused, you have the data to show where it's happening and you can adjust. You cannot audit an undocumented process. You can only audit a defined one.

9. Authority Boundaries

AI participant — pattern layer: - Receive and log the intake report - Provide the approved explanation of pending versus posted transactions - Gather transaction details (dates, amounts, reference numbers) from the customer - Create and route the intake record to the billing investigation queue - Send the customer an interim confirmation message with a reference number and investigation timeline - Answer general questions about the investigation process (timeline, what to expect) based on documented policy

Billing staff — authority layer: - Access the organization’s transaction records - Review the specific transactions at issue - Determine whether a duplicate charge occurred - Initiate a reversal if a duplicate is confirmed - Escalate to supervisor if the situation is ambiguous or if a provisional credit is warranted

Supervisor — elevated authority layer: - Authorize a provisional credit per the defined threshold policy - Communicate the investigation outcome to the customer - Handle exceptions (customer escalation, unusual transaction circumstances) - Document the investigation outcome in the case record

The critical boundary: the AI can explain what pending and posted mean in general terms. The AI cannot tell this customer that her specific transaction is definitely a duplicate or definitely not a duplicate. That determination requires access to actual transaction records and the authority to make a financial determination on behalf of the organization.

A second critical boundary: the AI cannot issue a provisional credit. This is not a pattern action — it is a financial commitment made on behalf of the organization, and it requires human authorization.

10. Stewardship Risks

Risk: The pending/posted explanation script becomes outdated. Banking practices, app presentations, and terminology drift over time. The explanation that is accurate today may be inaccurate in two years. The approved explanation script must be reviewed annually and whenever there is a change in payment processing systems or banking partners.

Risk: No standard script means no gap to find. If the pending/posted explanation is never standardized, there is no script to audit for drift. Stewardship cannot monitor what was never documented. The act of standardizing the explanation is also the act of creating something auditable. This must be established before stewardship is meaningful.

Risk: Provisional credit thresholds expand informally. Once a provisional credit process exists, there will be pressure to expand its use — a supervisor who wants to please customers, a staff member who tells callers “I’ll see if I can get you a credit.” Monitor the rate of provisional credit issuance monthly. If it is rising without a corresponding increase in confirmed duplicate charges, the threshold is being applied inconsistently.

Risk: Customer callback protocol degrades under volume. A callback commitment that is not completed within the promised window damages trust more than a longer initial promise. If call volume increases and the billing team cannot consistently meet the callback window, the stewardship response is to adjust the committed timeline — not to maintain the commitment and miss it repeatedly. Review callback compliance quarterly.

Risk: AI explanation creates over-reliance. If the AI explanation of pending versus posted is thorough enough, customers may accept it as a resolution rather than as an explanation. A customer who is told “pending transactions typically fall off within three to five business days” may stop following up, even if her specific transaction is a genuine duplicate that requires action. The workflow must ensure that

the explanation does not function as a substitute for investigation. Every intake with a specific billing concern must proceed to investigation, regardless of how satisfying the AI explanation was. # Chapter 19: Case Study — Bad Restaurant Experience

Overview

This case is structurally different from the two preceding it. The lost phone had a clear solution state: the phone in the customer’s hand. The double charge had a clear solution state: the correct amount on the account. This case has a solution state that is real but harder to name precisely — not a returned object, not a corrected number, but an experience made right.

What makes this case instructive is the emotional ambivalence of the caller. She is not primarily angry. She is disappointed. She does not want to cause trouble. She is uncertain whether she is entitled to complain at all. She explicitly asks whether she can “just let someone know.” These are not rhetorical moves or negotiation tactics. They are genuine expressions of a person who has been harmed mildly, who cares about fairness, and who is navigating the social discomfort of having something to say that she was not sure she was allowed to say in the restaurant.

The workflow must hold that complexity without collapsing it. The AI participant must receive the emotional disclosure, document the operational content, and route both — without over-promising, without under-responding, and without mistaking the caller’s ambivalence for satisfaction.

1. Messy Human Expression

The following is a near-transcript of a customer calling a restaurant after a difficult experience. She is calling the following day.

Phone rings. Staff member answers.

Caller: Hi — I’m calling about a dinner I had there last night? I almost didn’t call, honestly, because — I don’t know, I feel a little weird about it, but a friend said I should.

Staff: Oh, sure, of course. Can I ask what happened?

Caller: Well, it was — it was kind of a special occasion. My husband and I, it was our anniversary, so we — we don’t go out to nice restaurants very often, and we picked yours because we’ve always wanted to go. And I guess, um, the food was just — it took really long. We had an eight o’clock reservation and I think we didn’t get our food until after nine-thirty? Maybe nine-forty-five? And when it finally came, my entrée was cold. Like not a little cold. Cold. The pasta was — it was clearly not fresh, or it had been sitting, I’m not sure which. And I ate it because I didn’t want to make a big thing of it at the table, and it was our anniversary and I didn’t want to just, like, ruin the mood by complaining.

Staff: I understand —

Caller: And here's the thing — the server was really sweet. She was great, actually. I don't want her to get in trouble. I don't want this to be like, a complaint about her specifically, because she was very kind and she apologized, sort of, about the wait, she said the kitchen was backed up. So I understand that wasn't her fault. I'm not calling about her.

Staff: Okay, of course.

Caller: I just — I guess I want someone to know that this happened. Because we spent, I don't know, it was close to two hundred dollars, and it was our anniversary, and it was — it just wasn't what I expected from a restaurant at that price point. And I'm not — I'm not calling to get a refund or anything. I just — can I just tell someone? Is there a way to pass that along to a manager or whoever?

Staff: Absolutely, let me —

Caller: And actually — I mean, I guess I'm wondering — is there anything you can do? I don't want to sound like I'm asking for something, because that feels — I don't know. But is that something that's possible? Like a discount for a future visit or something? I'm not sure how that works. You don't have to. I just — I don't know, I thought I'd ask.

This transcript contains several layers operating simultaneously. There is an operational complaint: the food was late and cold. There is an emotional layer: it was a significant occasion, the disappointment was proportional to the anticipation. There is a social anxiety layer: she does not want to get the server in trouble, she is not sure she is entitled to ask for anything, she is qualifying her own complaint in real time. And there is an embedded request that she herself is not certain she has the right to make: she wants to know if a gesture is possible, but she does not want to be seen as someone who is asking for one.

2. Problem Statement

The customer had a meaningful dining occasion — an anniversary dinner — that failed on two operational dimensions: significant delay and a cold entrée. She did not raise the problem at the time, partly from social discomfort and partly from a desire not to compromise the occasion. She is now calling the following day, not primarily to demand redress, but to ensure that the experience is known and that the occasion can, in some meaningful way, be acknowledged as having mattered.

The full problem statement: a customer who invested emotionally and financially in a meaningful experience received a materially substandard version of it. She did not receive the opportunity to have the problem corrected in the moment because she chose not to raise it. She is now seeking acknowledgment that what happened was real, that the occasion had significance, and that the restaurant is willing to do something to honor the gap between what she expected and what she received. The gap between her

current state and resolution is: a genuine acknowledgment of the specific failures, confirmation that the feedback has reached someone with the authority to receive it meaningfully, and a gesture — not necessarily large — that signals the restaurant recognizes the occasion.

3. Solution Statement

The solution statement is: experience made right.

Experience made right does not require that the evening be retroactively restored. It requires that the customer ends the process feeling that the failure was received honestly, that the right person knows about it, and that some action has been taken that reflects the restaurant's recognition of the gap. A courageous and competent response acknowledges both the specific operational failures (the delay, the cold food) and the occasion context (the anniversary, the anticipation). A gesture — a credit, a complimentary return visit, a personal note from a manager — can constitute “made right” when it is specific to the experience and sincerely offered.

The solution statement is not “she was thanked for calling.” Thanking someone for calling is not resolution. It is reception. The solution requires action downstream of receipt.

4. Blind Ideal Simulation Summary

A well-configured AI participant receiving this call in demonstration mode will generate an idealized response sequence with the following shape:

The AI greets the caller and immediately signals genuine attention — not formulaic acknowledgment, but language that reflects that what the caller is describing has been heard. When the caller mentions the anniversary, the AI notes it explicitly: “I can hear that this was an important evening for you.” This is not flattery; it is the AI functioning as a bridge between the caller's experience and the organization receiving it.

The AI asks one or two clarifying questions to ensure the feedback record is specific: the approximate time of arrival, the name on the reservation, the nature of the entrée issue. It creates a structured feedback record that includes the occasion context, not just the operational facts. The record distinguishes the operational failure from the server relationship: the caller explicitly stated the server was not at fault, and the record reflects this.

The AI informs the caller that her feedback is being routed to a manager who will review it personally. It does not promise a specific outcome — no credit amount, no specific gesture — because those decisions require human authority. It does offer a specific follow-up: “A manager will review this and reach out to you within one business day.” If the restaurant has a standard courtesy response for experiences of this kind, the AI communicates what that process looks like without pre-committing the specific form.

The AI closes by thanking the caller in a way that reflects the weight of what she shared, not merely the fact that she called.

What the simulation reveals: the workflow requires a structured feedback intake form that captures occasion context, a routing mechanism to a named manager, a defined manager response window, a clear policy on what gestures are within the manager’s authority, and a follow-up mechanism that closes the loop with the customer.

5. Pure To-Be Model

Step 1 — Intake with occasion context The AI or staff member receives the call. A structured feedback record is created. Required fields: caller name, contact information, date of visit, reservation name, nature of the experience failure (delay, food quality, temperature, service), relevant context (occasion type, prior expectations, emotional register of the feedback), and explicit notes about personnel the caller has specifically excluded from the complaint.

Authority checkpoint: AI creates the feedback record. The occasion context — “anniversary, special visit, significant investment” — is a required field in the record, not optional color.

Step 2 — Acknowledgment within the call The AI or staff member acknowledges the occasion as well as the operational failure. This acknowledgment is not promotional language (“we pride ourselves on...”). It is specific to what the caller has described. If the caller mentions the server was kind, that is recorded and named in the response: “I’m noting that you’ve specifically said the server was attentive — that context will be included in the feedback.”

Authority checkpoint: Acknowledgment is a pattern action. The AI may acknowledge. The AI does not offer, promise, or hint at any specific remedy.

Step 3 — Routing to manager with context The completed feedback record is routed to a named manager — not “management,” not “the team.” A specific person is designated as the recipient of this type of feedback on this shift or day. The record includes the occasion context, the operational specifics, and the customer’s emotional register, so the manager is receiving the full picture, not just the complaint summary.

Authority checkpoint: Routing is AI’s responsibility. What happens next is the manager’s responsibility.

Step 4 — Manager review and response decision The manager reviews the record within the defined response window (recommended: within one business day for a next-day call of this type). The manager determines the appropriate gesture: a written acknowledgment, a credit, a complimentary return visit offer, or a personal phone call. This decision is the manager’s authority action.

Authority checkpoint: All gesture decisions — credit amount, comp offer, nature of follow-up — are manager authority. The AI has no role in this step beyond having routed the record accurately.

Step 5 — Customer follow-up The manager or designated staff member contacts the customer within

the stated window. The contact is personalized: it references the occasion, the specific failures named, and the server's positive contribution. The gesture offered, if any, is framed as recognition of the occasion gap — not as a standard complaint resolution.

Authority checkpoint: Customer follow-up is a human action. If the manager determines no gesture is warranted, the follow-up must still happen and must still be specific.

6. As-Is Discovery Questions

1. When a customer calls with a complaint about their dining experience, who typically takes that call? What is their training for handling it?
2. What information does the person taking the call actually write down? Is there a form, or is it informal?
3. Is there a distinction made between operational complaints (food quality, timing) and service complaints (staff behavior)? Does that distinction affect how the feedback is handled?
4. When a caller says “I don’t want to get anyone in trouble,” how does that typically affect what staff write down or pass along?
5. What happens to a complaint once it has been received? Where does it go and who looks at it?
6. Is there a defined window within which a manager reviews customer complaints? What is it?
7. Does the restaurant currently offer any standard gestures for experience failures — credits, comps, discounts — and if so, who decides when to offer them?
8. Is there a follow-up protocol for customers who call with feedback? Does anyone contact them after the complaint is logged?
9. When a customer says “it was a special occasion,” does that information get passed along to whoever handles the complaint, or is it treated as emotional context that doesn’t need to be documented?
10. Has there been a case where a customer called after a negative experience and the restaurant’s response turned them into a loyal customer? What made that work?
11. Has there been a case where a complaint was received but nothing followed — no acknowledgment, no gesture, no follow-up — and the customer left a public review? What did that look like?
12. Are there any cases where a staff member promised something (a credit, a future discount) without manager authorization? What happened?
13. Is the server mentioned in a complaint typically informed about the feedback? If so, how and by whom?

14. What does a manager do when they receive complaint feedback and they disagree that the restaurant was at fault — for example, if the kitchen delay was due to a large party and the timing was communicated to the server?
-

7. Likely As-Is Gaps

No occasion-context field in feedback intake. Feedback forms and complaint logs, where they exist, capture operational facts: what was wrong, when it happened, which table. They do not capture occasion context. A customer who was at an anniversary dinner receives the same intake treatment as a customer who was at a Tuesday lunch. This matters because the appropriate response differs materially. The workflow must treat occasion context as structured data, not narrative color.

Caller qualifications strip complaint severity from the record. When a caller says “I don’t want to get anyone in trouble,” front-line staff often interpret this as a directive to soften the complaint in the record. The result is a feedback record that documents “minor concerns about timing” instead of “ninety-minute wait and cold entrée on anniversary visit.” The manager who reads the softened record may decide no gesture is needed. Stewardship must protect the accuracy of complaint documentation against the social pressure of caller qualification.

No defined manager response window. Feedback is received but the timeline for manager action is undefined. Some complaints receive attention within hours; others sit in an inbox for a week. Without a defined window, the customer who was told “a manager will look at this” has no expectation to hold the restaurant to, and the restaurant has no standard to audit.

No standard gesture framework. The restaurant may have a cultural understanding that managers can offer something to unhappy customers, but without a defined framework — what gestures are available, at what level of complaint severity — managers make inconsistent decisions. One manager offers a full comp. Another offers nothing. The customer’s outcome depends on which manager received the record, not on the severity of the experience failure.

No follow-up mechanism. Many operations receive feedback and respond internally — the manager reads it, notes it, perhaps adjusts staffing or kitchen processes — without ever contacting the customer again. The customer who called “just to let someone know” receives no confirmation that anyone read the record, let alone acted on it.

8. To-Be Selling Conversation

The following dialogue excerpt shows the designer presenting the to-be model to a restaurant general manager.

Designer: When someone calls the morning after a bad experience, what's the most important thing that needs to happen?

Manager: We need to know about it. We need to make sure it gets to me, or to whoever is running that day.

Designer: And does it get to you consistently?

Manager: Honestly? Not always. It depends on who took the call and whether they thought to tell me.

Designer: So the gap isn't that your team doesn't care. The gap is that there's no structure guaranteeing the information moves. Someone could take a very well-handled call, hang up, and the feedback never reaches you because they got busy.

Manager: That's true.

Designer: What I'm proposing is a structured record. Not complicated — five fields. Caller name and contact, visit date, what specifically went wrong, occasion context, and what the caller said about the staff. Those five things go into a single form that every call-taker uses. The form routes directly to you. You don't find out from the hostess. You find out from the form.

Manager: And then what happens on my end?

Designer: You have a defined window — I'd recommend one business day — to read it and decide what, if anything, you want to do. And you have a short menu of options: a written acknowledgment, a credit of a defined amount, a complimentary return visit, or a personal call. You decide which one fits. But you make that decision within the window, and the customer gets contacted.

Manager: What if I think the complaint isn't that serious?

Designer: You still contact them. Even a sincere acknowledgment — not a credit, not a comp, just a personal note that says “your anniversary dinner should have been better and we know it” — closes the loop in a way that nothing else does. The risk in not responding is that the customer who called to let you know becomes the customer who posts a review to let everyone else know.

Manager: That's a fair point.

Designer: And one more thing. When a caller says “I don't want to get the server in trouble,” I want that in the record. I want the manager to know that the server behaved well and that the caller specifically excluded her from the complaint. That information matters for how you respond and for how you treat the server.

9. Authority Boundaries

AI participant — pattern layer: - Receive and document the feedback call - Capture occasion context as structured data - Acknowledge the experience and the occasion with specificity, using language calibrated to the emotional register of the caller - Note the caller's exclusion of specific staff from blame

- Create and route the structured feedback record to the designated manager - Communicate the follow-up timeline (“a manager will review this and contact you within one business day”) - Answer general questions about the feedback process based on documented policy

Staff receiving the call — support layer: - Ensure the caller feels heard before the call ends - Confirm that the record has been created and will be reviewed - Handle any escalation if the caller becomes more distressed during the call

Manager — authority layer: - Review the structured feedback record - Determine the appropriate response and gesture - Contact the customer within the defined window - Issue a credit, comp, or follow-up call as warranted - Communicate internally about any operational issue raised (kitchen delay, food temperature) - Determine whether and how the server is informed of the feedback

The critical authority boundary: the AI cannot issue a credit, offer a complimentary return visit, or promise that a manager will call by a specific time unless the organization has pre-authorized both the gesture and the timeline as standing policy. The AI can communicate policy. It cannot substitute for the manager decision that applies policy to a specific situation.

10. Stewardship Risks

Risk: Model drift causing under-reporting of complaint severity. A new AI model or a reconfigured prompt may interpret the caller’s language — “I don’t want to get anyone in trouble,” “I’m not asking for anything,” “I almost didn’t call” — as signals of low severity. A model calibrated for customer satisfaction scores may weight complaint severity against the caller’s expressed reluctance and generate a summary that reads as a mild concern rather than a meaningful experience failure. Monitor feedback records regularly: compare the customer’s stated experience against what the record documents. If records consistently under-represent the severity of calls, the AI configuration needs adjustment.

Risk: Follow-up protocol degrading under volume. A one-business-day response window works when complaint volume is manageable. During busy seasons, the volume may exceed the manager’s capacity to respond personally to each complaint within the window. Without a stewardship check, the window simply gets missed and no one notices. Audit follow-up completion monthly. If the window is being missed, either add capacity or adjust the committed timeline.

Risk: Gesture framework expanding informally. Once a gesture framework exists, there will be social pressure to expand it — a manager who wants to be generous, a caller who pushes, a staff member who tells a caller something is possible before checking. Monitor the rate of gesture issuance and the authorization pattern. Gestures issued outside the defined framework should be reviewed.

Risk: Server protection commitment not honored. When a caller explicitly states that a server should not face consequences for a complaint, that instruction must be tracked. If the server receives disciplinary attention connected to the complaint after the caller’s explicit request, the restaurant has broken a specific commitment. The feedback record must flag this instruction, and the stewardship process must confirm

that it was honored.

Risk: Feedback used for surveillance rather than improvement. Feedback records that reach management can be used to improve operations or to monitor staff. When staff become aware that complaints flow to managers, they may alter their behavior in ways that suppress complaint reporting from callers — discouraging calls, softening records. The intent of the feedback workflow is operational improvement. Stewardship must ensure it is not functioning as a performance surveillance mechanism, which would undermine staff trust and ultimately reduce the quality of intake. # Chapter 20: Case Study — Sales Call to Decision Maker

Overview

Every case study in this section has involved a customer seeking resolution from an organization. This case inverts the direction of contact. Here, the person initiating the interaction — the salesperson — is the one seeking something from the organization being contacted. The contacted organization’s AI participant is not serving a customer in distress; it is operating as a gatekeeper and qualifier between an incoming opportunity and the human decision authority that lives inside the organization.

This distinction is not trivial. It changes everything about what the AI participant is doing and what success looks like. In the lost-phone or double-charge case, the solution state is relatively clear: the phone is found, the charge is corrected. In the sales call case, the solution state is one of two things — neither of which is “the call happened.” The solution state is either a qualified path to a decision maker, or a respectful disqualification. Both are valid resolutions. A pleasant conversation that produces neither is false completion, and false completion in this case is costly for both parties.

1. Messy Human Expression

The following is a near-transcript of a salesperson calling a business and encountering an AI-assisted intake participant. The salesperson has been calling this type of company for several years and has developed a practiced cadence, but their underlying objective — access to someone who can make a purchasing decision — is partially obscured by that cadence.

Phone connects. AI participant answers.

AI: Thank you for calling Meridian Group. How can I help you today?

Salesperson: Hi, yes — my name is Tom Castillo, I’m with Halford Business Solutions. I’m calling because we work with companies in your space and we’ve had some really strong results helping operations teams streamline their vendor management process. Is there someone I could speak with about that?

AI: Of course. Can I ask a bit more about what you’re looking for?

Salesperson: Sure, absolutely. We've developed a platform that integrates with most existing ERP systems — SAP, Oracle, some of the mid-market stuff — and basically cuts the time your procurement team spends on vendor onboarding and compliance tracking by something like forty percent. We've got a few case studies I could share. I really just want to get five minutes with whoever handles your operations or procurement function. Is that someone you can connect me with, or would it make more sense to schedule a call?

AI: I appreciate the overview. To make sure I'm routing you to the right person, can I ask — is this something you've been exploring with Meridian Group before, or is this an initial outreach?

Salesperson: This is initial, yeah. Cold call, if we're being honest. But it's a good fit — I've done my research on the company, you're in the range where this kind of platform makes a real difference. I really just need five minutes.

AI: Understood. And do you have a sense of what timing looks like on your end — are you looking to have a conversation this quarter, or is this more exploratory?

Salesperson: Honestly, I'm ready to move whenever it makes sense for the other person. We have capacity right now. So if there's someone who handles this and they're even a little curious, I'd love to get on their calendar.

AI: Got it. Let me — is there anything specific that's prompting this call right now? Like, are you responding to something you saw about Meridian, or is this a standard outreach to companies in this sector?

Salesperson: Standard outreach. We identify companies that fit our profile and we call them. That's the honest answer. I'm not going to pretend otherwise. But I'm also not going to waste anybody's time — if you can tell me there's genuinely no one here who handles procurement decisions, I'll mark it as not a fit and we're done. I just need to know I'm talking to the right person, or that I should find another way in.

This transcript has a quality that distinguishes it from the others in this section: the salesperson is practiced, self-aware, and ultimately requesting something very specific — a decision signal. He is asking: “Is there qualified opportunity here?” His language near the end of the excerpt — “if you can tell me there's genuinely no one here who handles procurement decisions, I'll mark it as not a fit and we're done” — is not a negotiating move. It is a genuine invitation to produce a clean disqualification if one is warranted. This is a salesperson who understands that a clear no is more useful than an indefinitely warm non-response.

The AI participant in this call has a significant responsibility. It is not here to deflect, stall, or produce ambient friendliness. It is here to determine whether a qualified opportunity exists and, if so, to route it; or to produce a clear, respectful disqualification if one does not.

2. Problem Statement

The salesperson needs to determine whether a qualified purchasing opportunity exists at Meridian Group — meaning: is there a person with budget authority, an identified need in the salesperson’s solution area, and a realistic timeline for a decision? Without this determination, the salesperson cannot advance the opportunity or release it. He is currently in a state of unknown — which is not the same as a no, but is also not a path forward.

The full problem statement: a salesperson with a potentially relevant solution is seeking access to a decision maker at a target organization. He cannot determine from the outside whether decision authority exists, whether budget is allocated for this category, or whether there is an active or latent need. His time and the organization’s time are both at risk of being consumed by a process that produces neither a qualified advance nor a clean disqualification. The gap between his current state and resolution is: a determination of whether a qualified opportunity exists, and if so, a defined path toward it.

3. Solution Statement

The solution statement is: qualified path to decision maker, OR respectful disqualification.

These are equivalent resolutions. A qualified path means: the salesperson has been connected to, or has a confirmed appointment with, a human who has the authority to evaluate and act on a purchasing decision in this category. The conversation that follows may or may not result in a sale — that is beyond the scope of this workflow — but the path itself is defined and real.

A respectful disqualification means: the salesperson has been told clearly, by someone or something with enough organizational knowledge to say so, that there is no active need, no budget allocated, or no decision authority present for this category at this time. This is not a brush-off — it is a resolution. It closes the opportunity cleanly for both parties and is, in every meaningful sense, as valuable as a qualified advance.

The solution state that must not be allowed to masquerade as resolution: a warm, pleasant AI conversation in which the salesperson is thanked for their call, told that “this will be passed along,” and given no decision signal. This produces a false completion. The salesperson leaves the call feeling that something happened. In fact, the opportunity is now in an undefined state from which it is unlikely to advance, and the salesperson has no accurate information to act on.

4. Blind Ideal Simulation Summary

A well-configured AI participant receiving this call in demonstration mode will generate an idealized response sequence roughly as follows:

The AI greets the caller, accepts the introduction, and moves quickly to qualification. It does not ask for a product overview — it already has enough to know what kind of call this is. It asks the caller a defined set of qualification questions drawn from standard sales qualification frameworks: Is there an existing vendor or system in this category? Is this category currently under active evaluation? What would be the approximate timing for a decision if interest were confirmed? Has the organization previously engaged with vendors in this space?

The AI listens to the salesperson’s answers and categorizes the call accordingly. If the salesperson’s fit signals are strong — the category is active, there is a known decision maker, the timeline is real — the AI routes the call to the appropriate internal contact or offers to schedule a call with that person. If the fit signals are weak or ambiguous, the AI thanks the caller genuinely, provides a specific reason why this is not the right time (if one exists in the organization’s records — for example, “we are mid-contract with a current vendor in this category until Q3”), and offers a concrete path for re-engagement if circumstances change.

The AI does not conduct a sales conversation. It does not engage with the salesperson’s product features, case studies, or ROI claims. Its role is qualification and routing — pattern recognition that produces either a path or a disqualification.

What the simulation reveals: the workflow requires a defined set of qualification questions, a routing decision tree, authority to represent the organization’s current state (existing vendor relationships, budget allocation, active evaluations), and a disqualification script that is specific enough to be useful.

5. Pure To-Be Model

Step 1 — Call identification The AI identifies the nature of the incoming call within the first exchange. A vendor or sales outreach call is distinguishable from a customer support call or an inquiry from an existing partner. The identification may be based on caller introduction, call source, or explicit declaration.

Authority checkpoint: AI identifies and categorizes. It does not decide whether to engage; all properly identified sales calls enter the qualification process.

Step 2 — Structured qualification The AI conducts a structured qualification sequence using a defined set of questions. The questions are not improvisational. They are designed to determine: Is there a named role or department responsible for decisions in this category? Is this category currently under active evaluation or contracted? What is the realistic decision timeline? Has this vendor or a similar vendor engaged with this organization before?

Authority checkpoint: AI conducts the qualification. AI does not answer questions about the organization’s internal operations beyond what it has been explicitly authorized to represent. If the salesperson asks “are you currently using a competitor’s product,” the AI can confirm or deny based on authorized information; if the information is not authorized, the AI routes to a human.

Step 3 — Routing decision Based on the qualification results, the AI makes one of three determinations: (a) Qualified opportunity — route to designated decision maker or schedule a call with that person; (b) Possible opportunity — route to a gatekeeper who can make the final routing decision; (c) Disqualification — provide the salesperson with the appropriate reason and close the call.

Authority checkpoint: Routing is an AI pattern action. The decision maker’s calendar access, the gatekeeper’s authority to accept or reject the routing, and the content of the disqualification reason are all established in advance by human authority. The AI executes within those parameters.

Step 4 — Advance or disqualification If qualified: the AI confirms the routing. “I’m connecting you with [role name], or offering a scheduled call. Here is what they will need from you to prepare.” If disqualified: the AI provides the specific reason, thanks the salesperson specifically and briefly, and closes the call. The disqualification is logged.

Authority checkpoint: Scheduling a call requires access to the decision maker’s calendar, which must be pre-authorized. Representing a disqualification reason requires that the organization has pre-approved the language used.

Step 5 — Log and follow-up All sales call intakes are logged with: caller name, company, stated product category, qualification signals, routing decision, and outcome. Logs are reviewed by the relevant department head on a defined schedule to ensure that the qualification criteria remain current.

Authority checkpoint: Log review is a human responsibility. The criteria used by the AI for qualification must be validated periodically by the decision makers who would ultimately take the advanced calls.

6. As-Is Discovery Questions

1. When a salesperson calls, who typically takes that call? Is there a consistent intake role, or is it whoever answers the phone?
2. What does the person answering do in the first thirty seconds of a sales call? Is there a protocol, or is it improvised?
3. Is there a defined set of decision makers for different purchasing categories — for example, one person for technology, one for facilities, one for professional services? Is that structure documented?
4. When a salesperson asks to speak to “whoever handles procurement,” what typically happens? Is there a consistent routing path?
5. Are there any purchasing categories where an active evaluation is ongoing right now? If so, is the AI participant authorized to represent that?
6. Are there any categories where the organization is mid-contract and not looking to evaluate vendors? Is the AI authorized to say so?

7. How does the organization currently handle sales calls that seem relevant — are they logged, forwarded, discarded?
 8. Has a useful vendor relationship ever originated from a cold inbound sales call? If so, what made that routing work?
 9. Has a sales call ever been routed to the wrong person or misdirected in a way that created problems? What happened?
 10. Is there currently any guidance — formal or informal — for staff on how to qualify a sales call before routing it?
 11. When a sales call is disqualified, what does the caller typically get told? Is there consistency in that message?
 12. Do decision makers in the organization currently receive unsolicited sales inquiries that they find irrelevant? What do they say about that?
 13. Is there a way for a disqualified caller to re-engage at a future point — a defined path for “call back in six months” — or is disqualification a permanent close?
 14. Are there any legal or compliance constraints on what the AI can represent about the organization’s vendor relationships, budget, or procurement processes?
-

7. Likely As-Is Gaps

No qualification structure. Most organizations that receive inbound sales calls handle them through whoever answers — usually a receptionist or general customer service representative — who has no training in sales qualification. The call is either routed immediately (producing an unwanted interruption for a decision maker) or deflected with vague language (“I’ll pass your information along”), which produces false completion for the salesperson and no useful outcome for the organization.

No routing decision tree. Without a documented mapping of who makes decisions in which categories, routing is based on guesswork. “Whoever handles purchasing” might be the CFO or the office manager, depending on who is asked.

No disqualification language. Organizations rarely train anyone on how to disqualify a sales call clearly and respectfully. The most common response — “let me take your information” — is neither a qualification nor a disqualification. It produces a follow-up obligation the organization will not honor and a false signal to the salesperson.

No log of incoming sales calls. Without a log, the organization cannot identify which vendors have called, how often, what they are offering, or whether any of those offerings might have been relevant at a different time. A sales call that was disqualified in January because no budget existed might have been a strong fit in March after a budget cycle. Without a log, that connection cannot be made.

Decision maker contact information not protected. In many organizations, decision makers' direct lines or emails are easily discoverable, so the qualification step is bypassed entirely. The AI's role in protecting decision maker time depends on the organization having established that the AI is the intake point for sales calls, not a waystation before the caller finds another path in.

8. To-Be Selling Conversation

The following dialogue shows the designer presenting the to-be model to an operations director who currently receives too many unqualified sales calls.

Designer: How many sales calls do you handle in a given week?

Director: Too many. Probably five to ten that get through to me. Most of them aren't relevant.

Designer: And how do they get to you?

Director: Someone transfers them. Or they look up my name and call directly.

Designer: What if none of them reached you unless they had already been qualified?

Director: That would be ideal. But who does the qualifying?

Designer: The AI participant does the first pass. It has a set of defined questions — four or five — that identify whether the call is in an active category, whether there's any possibility of budget, and what the timing looks like. If those signals are strong, it routes to you or schedules a call on your behalf. If they're weak, it disqualifies cleanly and closes the call.

Director: What if the AI gets it wrong? What if something sounds weak but it's actually interesting?

Designer: Two things. First, the criteria are set by you — you decide what signals indicate a qualified call. Second, every call is logged. If something gets disqualified and later you hear about a vendor who you wish you'd talked to, you can pull the log and find the call. The log is the safety net.

Director: What about the salesperson? Is it going to be awkward for them?

Designer: Only if the disqualification is vague. A clear, specific reason — “we're under contract in this category until Q4 of next year, and we're not evaluating at this time” — is more respectful than “I'll pass your information along.” The salesperson gets a real answer. They can plan around it. A real no is better for everyone than an indefinite maybe.

Director: And what do I actually see when something is routed to me?

Designer: A record. Date of call, caller name, company, product category, the answers they gave to the qualification questions, and a brief summary. You're not walking into a cold call. You're walking into a documented introduction.

9. Authority Boundaries

AI participant — pattern layer: - Identify the call as a sales outreach - Conduct the structured qualification sequence - Represent the organization's publicly known or pre-authorized information about its current procurement state - Produce a routing decision based on qualification signals - Log the call with full details - Deliver the disqualification script when warranted - Offer scheduling access for qualified calls if pre-authorized calendar access has been granted

Human gatekeeper — intermediate authority layer: - Review ambiguous routing decisions before they reach the decision maker - Override an AI disqualification if there is context the AI lacks - Approve access to the decision maker's calendar for scheduled calls

Decision maker — authority layer: - Accept or decline the routed call - Validate that the qualification criteria used by the AI remain current - Determine whether a disqualified category should be reopened - Engage substantively with the salesperson if a call is advanced

The critical authority boundary in this case is commitment of decision maker time. The AI can offer a scheduled call only if it has explicit pre-authorization to access the decision maker's calendar and to make that offer on behalf of the organization. Without that authorization, the AI can tell the salesperson that a qualified call will be arranged — but the actual scheduling is a human step. A false scheduling commitment — “I'll have someone reach out to set up a time” followed by no follow-up — is not a neutral error. It is a broken promise.

A second boundary: the AI cannot represent the organization's budget status, contract terms, or procurement authority structure unless those representations have been explicitly authorized. Representing that “we are under contract until Q4” is an organizational statement that requires authorization. If that information has not been provided to the AI, the AI must route to a human who can confirm or deny.

10. Stewardship Risks

Risk: AI becomes too warm, producing pleasant but unresolved outcomes. An AI participant that is configured for customer warmth and satisfaction may apply those same qualities to sales call handling in ways that produce false completion. A salesperson who receives a warm, engaged conversation, is told “this sounds interesting, I'll make sure someone follows up,” and then receives no follow-up has been harmed by the AI's warmth. Monitor not just call completion but the rate of qualified advances and clean disqualifications. If the majority of sales calls end in “I'll pass this along” without a routing action or a logged disqualification, the AI is producing ambient friendliness and not resolution.

Risk: Qualification criteria become outdated. The criteria used to qualify or disqualify a sales call reflect the organization's procurement state at the time the workflow was designed. If a budget cycle

changes, a contract expires, or a new category opens up, the qualification criteria must be updated. A call that should now be qualified will be disqualified if the criteria are stale. Review qualification criteria quarterly with the relevant decision makers.

Risk: Disqualification reason becomes too generic. If the disqualification script defaults to generic language — “we’re not looking at this right now” — it loses its value. Salespeople learn to ignore it, and the organization loses the reputational benefit of a clear, respectful interaction. Review disqualification language annually and calibrate it to the actual reasons calls are being declined.

Risk: Log review stops happening. The sales call log is designed to provide organizational intelligence about what kinds of outreach the company is receiving and whether the qualification criteria are working. If log review stops — because it is nobody’s specific responsibility — the log becomes an archive rather than a tool. Assign a specific role and a defined review frequency. Review outcomes: what percentage of qualified advances resulted in productive conversations? What percentage of disqualifications were correct?

Risk: Salespeople find alternate paths around the AI intake. If a salesperson is disqualified by the AI and then calls back, asks to speak to a specific person by name, or emails a decision maker directly, the AI’s role is circumvented. This is not primarily a technology problem; it is an organizational one. Decision makers must understand that the intake exists, trust it to route qualified calls accurately, and have a protocol for handling direct outreach that bypasses it. If they accept every call that comes in directly, the intake workflow is decorative.

Risk: False completion becomes normalized. The most insidious stewardship risk in this case is cultural. If the organization’s AI participant consistently produces warm, friendly sales call interactions that end without a routing decision or a clean disqualification, and if no one is auditing the outcomes, that pattern becomes the norm. Staff begin to see “she handled it professionally” as success, regardless of whether any actual routing occurred. The stewardship process must insist on outcome metrics: not how well the call went, but what happened at the end of it. # Chapter 21: Case Study — Business Idea to Communicable Offer

Overview

Every case study in this section has involved an external contact — a customer, a caller, a salesperson — and an organization with an existing operational structure receiving that contact. This case is different in kind. There is no transaction to resolve, no charge to correct, no complaint to route. There is a founder and an idea that the founder cannot yet put into words that another person can understand, respond to, or act on.

This is the most interior case study in the book, and it requires the most careful framing. The problem is not that the idea is bad. The problem is not that the founder lacks intelligence or commitment. The problem is that the idea has not yet been translated into a form that can survive contact with a listener who has no prior knowledge of it. Until it can, the idea cannot spread, cannot be tested, and cannot become an offer.

The AI participant in this case functions as something genuinely unusual: a structured listener. Not a validator. Not a market analyst. Not a business coach. A structured listener whose primary function is to receive the full unfinished expression, identify the moments of near-clarity within it, reflect back a structured form, and iterate until the founder can say “yes — that is what it is.”

The workflow designer’s task is to build the conditions under which that iteration can happen reliably — and to ensure that the AI’s role ends where authority begins.

1. Messy Human Expression

The following is a near-transcript of a founder attempting to explain their business idea to someone they have just met. The founder has been working on this for eight months. They believe in it. They have not yet found words for it that work.

Founder: So, basically — okay, so you know how when you’re a small business, like, a really small business, not a startup with funding but like, a real small business, like a bakery or a contractor or whatever — you know how they always have like seventeen different apps they’re supposed to be using? There’s one for scheduling, one for payments, one for invoicing, one for customer communication, and none of them talk to each other?

Listener: Yeah, that sounds familiar.

Founder: Right, so — that’s the thing I’m trying to fix. But it’s not just an app, it’s more like — it’s more like a layer that sits on top of all of that. So it’s not replacing anything, it’s just — it’s like the thing that ties them together. But also it’s not just integration software, because that’s kind of a hard sell to someone who doesn’t know what an API is, you know? It’s more like — it’s more like you have an assistant who knows all your systems. Does that make sense?

Listener: Sort of. So it’s like a dashboard?

Founder: Not really a dashboard. I mean, there’s a dashboard component, but — it’s more conversational? Like you could ask it a question and it would pull from all your different systems to answer it. Like if you ask “who are my best customers this month” it would pull from your sales system and your communication history and give you an actual answer, not just a number. So it’s more like — it’s like a business interpreter. That’s actually a phrase I’ve been using internally but I’m not sure it means anything to someone who hasn’t heard my whole pitch.

Listener: What kind of businesses is this for specifically?

Founder: That’s the thing — it depends on who I’m talking to. Like, if I’m talking to a contractor, I lean into the scheduling piece. If I’m talking to someone in food service, I lean into the customer communication piece. So it kind of shifts. Which I know is not great, I know I need a cleaner answer, I just haven’t nailed it yet. I think the real customer is anyone who is running a business with fewer than

fifteen employees and feels like they're managing their tools instead of running their business. That's the feeling I'm trying to solve. The feeling of being managed by your software. But that's also not a very concrete pitch.

Listener: No, I mean, I kind of get it. It sounds interesting.

Founder: Yeah, people usually say that and then they don't really — I can't tell if they actually get it or they're being polite. Which is honestly why I'm still working on how to explain it.

This transcript is a founder at a specific and recognizable stage: they know what they are building, they can describe it from multiple angles, and they have tried several framings — layer, assistant, dashboard, business interpreter, tool-management-fatigue solver. None of the framings have landed cleanly. The listener's response — “I mean, I kind of get it. It sounds interesting” — is the canonical polite non-engagement that signals the idea has not yet crossed the communicability threshold.

The phrase the founder uses near the end — “I can't tell if they actually get it or they're being polite” — is the clearest articulation of the problem. It names the gap precisely: there is no signal confirming that the idea has transferred.

2. Problem Statement

The founder has built something real. She has a functional or near-functional product, a real customer pain point she has identified, and at least one framing — “you feel like your tools are managing you instead of you managing them” — that comes close to landing. But when she explains what she is building in conversation, she cannot determine whether the listener has understood it. Listeners say “that sounds interesting” and then do not engage further, do not ask follow-up questions that demonstrate comprehension, and do not say “I know someone who needs this” or “I would pay for that.”

The full problem statement: the founder has a real offer that cannot be transmitted. In its current form, the explanation wanders through multiple framings without settling on one, requires context the listener does not have, and does not answer the listener's implicit questions: what is it, exactly? Who is it for? What do I do next? The gap between the current state and resolution is: a single clear expression of the offer that a stranger — someone who has never heard of this product — can understand, respond to, and potentially act on after hearing it once.

3. Solution Statement

The solution statement is: idea made communicable.

Communicable means a specific and testable standard, not a feeling. It does not mean “sounds good to me.” It does not mean “the founder likes how it sounds.” Communicable means: when the founder speaks the offer to a stranger who has never heard of it, that stranger understands what the product is, who it is for, and what happens next. The stranger should be able to repeat the core of it back without prompting. If they cannot — if they say “so it’s kind of like a dashboard?” or “interesting, I’d have to think about that” — the offer has not yet cleared the communicability threshold.

The solution state is also not permanent. As the business grows, as the product develops, as the founder’s understanding of the real customer deepens, the communicable offer will need to be renewed. The workflow described here produces a version that is communicable at this stage of development. Stewardship is the process of keeping it current.

4. Blind Ideal Simulation Summary

A well-configured AI participant working with this founder in demonstration mode will generate an idealized interaction sequence roughly as follows:

The AI invites the full expression without interruption. It does not ask clarifying questions during the initial expression; it receives the whole thing. After the founder has finished — which may take several minutes and include tangents, restarts, and pivots — the AI produces a structured reflection: “Here is what I heard.”

The reflection takes a specific form. It names three things: what the product is (the one-sentence operational description), who it is for (the specific person or role who experiences the pain it addresses), and what happens next (the concrete action a prospective customer would take). The AI presents this not as a conclusion but as a draft: “Is this close?”

The founder responds. Perhaps one element is right and another is off. The AI adjusts the reflection and presents a revised version. This iteration continues — typically three to five rounds in a demonstration — until the founder says: “Yes. That’s it.”

The AI then asks one additional question: “Would a stranger who knows nothing about this understand that explanation?” If the founder hesitates, the AI prompts one more iteration. If the founder says yes confidently, the AI produces a final written version of the offer expression: the short form (one to two sentences), the medium form (a brief paragraph), and the conversational form (how to open it in a networking situation).

What the simulation reveals: the workflow requires a structured reflection protocol, a defined communicability standard, and an iteration process that is bounded — it has an ending condition, not an endless loop. It also requires a clear boundary: the AI’s job is to reflect and structure. The AI does not validate the business idea, confirm market fit, or tell the founder the offer will work. Those are authority questions that require human judgment, market data, and domain expertise the AI does not have.

5. Pure To-Be Model

Step 1 — Full expression The AI invites the founder to explain the business idea in whatever form is natural, without interruption. The AI’s posture in this step is receptive: it is holding space for the full expression, including tangents, restarts, and competing framings. The goal of this step is not to gather specific information — it is to hear the whole thing before attempting to structure it.

Authority checkpoint: No structuring, evaluating, or redirecting occurs in this step. The AI listens.

Step 2 — Identification of near-clarity moments Before reflecting, the AI internally identifies the moments in the founder’s expression where the explanation came closest to clarity — where the language was most specific, most grounded in a real person’s experience, most likely to land with a stranger. In the transcript above, the phrase “you feel like your tools are managing you instead of you managing them” is a near-clarity moment. So is the description of a small business owner with fewer than fifteen employees who feels overwhelmed by software fragmentation.

Authority checkpoint: The AI surfaces these moments in its reflection. It does not suppress them in favor of a cleaner-sounding summary that loses their texture.

Step 3 — Structured reflection The AI produces the “here is what I heard” reflection. The structure is fixed: (1) What is the product — the simplest operational description, (2) Who is it for — the specific person experiencing the pain it addresses, (3) What happens next — the concrete action a prospective customer would take. The AI presents this as a draft, not a conclusion.

Authority checkpoint: The reflection is accurate to what the founder said. The AI does not add market analysis, competitive framing, or validation. It reflects. The founder evaluates.

Step 4 — Iteration The founder responds to the reflection. If it is close but not right, the AI adjusts the relevant element and presents a revised version. Each revision cycle addresses one element at a time: what it is, who it is for, or what happens next. The iteration has a defined ending condition: the founder confirms that the reflection is accurate and that the language is clear.

Authority checkpoint: The AI iterates based on the founder’s feedback. The AI does not substitute its judgment for the founder’s on what the offer actually is. If the founder says “no, that’s not quite right,” the AI asks “what would you change?” and works from the founder’s correction, not from its own inference.

Step 5 — Communicability test Before producing the final version, the AI asks: “Would someone who has never heard of this understand this in one hearing?” If the founder hesitates, the AI prompts one more iteration. If the founder confirms, the AI produces the three-form written version: short form, medium form, and conversational form.

Authority checkpoint: The communicability test is the founder’s judgment, not the AI’s. The AI prompts the test; it does not pass or fail the offer on the founder’s behalf.

Step 6 — Written output The AI produces the written output. The founder reviews it. If anything has been lost in the transition from conversation to text, the founder corrects it. The written version is treated

as a working document, not a final artifact: it will be used, tested in real conversations, and updated as the founder's understanding evolves.

Authority checkpoint: The written output belongs to the founder. The AI has no claim on it, no authority to declare it finished, and no ability to validate whether it will work in market.

6. As-Is Discovery Questions

In this case, the “as-is” is the founder's current explanation practice — how they currently try to communicate the idea, what happens when they do, and what feedback they have received.

1. When you explain your business to someone new, how do you usually start? What's the first sentence you typically say?
2. How long does your explanation usually take before you feel like the other person has understood it?
3. What's the most common response you get? What do people say — or not say — that tells you whether they got it?
4. Have you ever explained it to someone and felt that they genuinely understood it? What did they do or say that made you feel that way?
5. Have you tried writing it down? If so, does the written version feel accurate when you read it back?
6. When you're talking to different kinds of people — potential customers, investors, partners — do you explain it differently? What changes?
7. Is there one version of the explanation that you keep returning to — one framing that feels closest to right, even if it doesn't always land?
8. What do you think is hardest to explain about this? What part always requires extra clarification?
9. When someone says “that sounds interesting,” do you have a sense of whether they actually understood it, or whether they're being polite?
10. Have you gotten feedback — from customers, from pilots, from early users — that changed how you think about what the product actually is?
11. What would a customer say to someone else about this? How would they describe it if you weren't in the room?
12. Is there a person who you feel has understood this most completely? What did you tell them? What did they say back?
13. What does it feel like when the explanation doesn't land? Where in the conversation does it usually go sideways?

14. If you could only say one sentence about this to someone and you had to make it clear, what would that sentence be?
-

7. Likely As-Is Gaps

Multiple framings in active competition. The founder has tried several framings — layer, assistant, dashboard, business interpreter, tool-management fatigue — and none has been retired. They coexist and compete within the same explanation. This produces confusion for the listener, who cannot determine which framing is the actual one. The gap is not that the framings are bad; it is that there is no selection process. The communicability work is partly a selection process: which framing carries the most of the idea with the least effort from the listener.

The listener’s implicit questions go unanswered. Every new listener of a business idea is asking three questions, even if they do not ask them aloud: What is it? Who is it for? What do I do next? The founder’s current explanation addresses what it is — imperfectly, through multiple framings — but does not clearly answer who it is specifically for or what the listener should do if they are interested. An explanation that ends without a clear next step leaves the listener’s implicit question unanswered, and “that sounds interesting” is the standard response to an unanswered question.

No feedback loop from failed explanations. The founder notices that explanations are not landing — “I can’t tell if they actually get it or they’re being polite” — but has no structured process for capturing why they failed. Which framing was in use? What did the listener say? At what point in the explanation did engagement fade? Without this data, each explanation attempt is disconnected from the previous one, and improvement is slow and accidental.

Communicability confused with comprehensiveness. Founders often believe that a longer, more comprehensive explanation will produce more understanding. The opposite is frequently true. A comprehensive explanation of a multi-component product — scheduling, payments, invoicing, communication, integration layer, conversational interface — asks the listener to hold more information than they can process in a first encounter. Communicability is achieved through reduction, not addition. The as-is gap is the assumption that more explanation equals more clarity.

No defined end condition for the explanation. The founder does not have a signal for when the explanation is done. It ends when the listener changes the subject, or when a natural pause occurs, or when the founder runs out of framings. None of these are communicability signals. The to-be workflow requires a defined end condition: the listener can say back what the product is, who it is for, and what happens next.

8. To-Be Selling Conversation

In this case, the “selling conversation” is between the workflow designer and the founder. The designer is proposing that the founder accept a specific and rigorous standard for what “communicable” means — and that the process of reaching that standard requires structured work, not simply continued practice.

Designer: When you explain this to someone and they say “that sounds interesting,” what do you take away from that?

Founder: I used to take it as a positive response. Now I’m not so sure.

Designer: What changed?

Founder: Nothing happened after. Like, no one followed up. No one asked me how to learn more. It just — the conversation moved on. And I started to think “sounds interesting” might be polite for “I didn’t really understand that.”

Designer: It often is. “Sounds interesting” is what people say when they don’t want to say “I don’t know what you just described.” It’s a graceful exit from a conversation that didn’t land.

Founder: So what’s the alternative? What does it look like when it does land?

Designer: The listener asks a specific follow-up question. Not a clarifying question — not “wait, so what does it actually do?” — but a forward question. “Do you work with businesses in my industry?” “How much does it cost?” “Who would I talk to about piloting this?” Those questions mean the idea crossed over. The listener understood it well enough to ask what comes next.

Founder: I want that.

Designer: Right. And here’s what I need you to accept as the standard we’re working toward. “Communicable” does not mean it sounds good to you, or it sounds good to me, or it sounds polished. It means a stranger who has never heard of this understands it in one hearing and knows what to do next. That is the test. Not whether it’s a good idea — I have no standing to tell you that. Not whether it will work in market — I cannot tell you that. Those are real questions, but they are not the question we’re answering here. The question we’re answering is: does the explanation transfer?

Founder: And you think it doesn’t yet.

Designer: I think you have the raw material. I can hear the near-clarity in some of what you said. “You feel like your tools are managing you instead of you managing your business” — that sentence is doing work. It names the pain in the customer’s own experience. That’s the beginning of a communicable offer. What we need to do is build the rest of the sentence, the sentence before it, and the sentence after it, so that together they produce a complete transfer.

9. Authority Boundaries

AI participant — pattern layer: - Hold the full expression without interruption - Identify near-clarity moments within the expression - Produce a structured reflection (“what it is,” “who it is for,” “what happens next”) - Iterate the reflection based on founder feedback - Apply the communicability test (“would a stranger understand this in one hearing?”) - Produce the written output in short, medium, and conversational forms - Prompt renewal when the founder indicates the offer has evolved

Founder — authority layer: - Confirm or correct the AI’s reflection - Decide which framing is the actual one - Determine whether the communicability test has been passed - Approve the written output - Deploy the offer in real conversations and report back on what happened

Domain expert or market advisor — external authority: - Validate whether the offer addresses a real market need - Confirm whether the described pain exists at the scale and frequency the founder believes - Assess competitive landscape - Evaluate pricing, positioning, or go-to-market strategy

The critical authority boundary in this case is validation. A founder who has worked with the AI to produce a clear, communicable offer may leave the process with more confidence than the process warrants, if they conflate communicability with viability. The AI has helped them say what the offer is clearly. It has not confirmed that anyone will buy it. It cannot confirm that. That determination requires human judgment, market research, and tested customer conversations.

This boundary must be stated explicitly. The AI’s closing statement in a communicability workflow should include something like: “This is a clear expression of your offer. Whether the offer itself will find market is a different question, and one that requires real customer conversations, not further iteration with me.”

A second critical boundary: the AI cannot tell the founder the offer is good. It can tell the founder the offer is clear. These are different statements. Clarity is a structural property. Goodness is a market judgment. The AI has no standing to make market judgments.

10. Stewardship Risks

Risk: The offer changes but the expression does not. Businesses evolve. Founders learn. Early customers reveal that the real value is different from what was initially described. When this happens, the communicable offer that was developed through this workflow becomes inaccurate. An expression that was once clear now obscures because what it is describing is no longer what the business is doing. Stewardship requires the founder to revisit the communicability workflow whenever the underlying offer changes substantially. Renewal is not failure; it is the natural lifecycle of a living business.

Risk: Communicability mistaken for market validation. This risk has been named in the authority boundaries section, but it is significant enough to warrant explicit stewardship attention. A founder who has a clear offer may begin to treat clarity as evidence of fit. They may say “I’ve got my pitch dialed in”

as if that means customers are ready to buy. Monitor the feedback from real conversations: is the new expression producing forward questions? Is it producing follow-up? If clear expression is not producing engagement, the issue may not be communicability — it may be offer design, pricing, target customer selection, or channel. Those issues are downstream of clarity and require different kinds of work.

Risk: Model drift causes over-smoothing. A language model configured to produce well-formed prose may, over time, produce reflections that are cleaner than the founder’s actual idea. The reflection sounds right — it is grammatical, structured, and polished — but it has subtly replaced the founder’s rough near-clarity with language that sounds professional but does not actually represent what the founder is building. The founder may accept it because it sounds better, not because it is more accurate. Monitor for this by asking the founder to explain the offer in their own words after reviewing the AI’s written output. If they cannot reproduce the substance of the written version in their own language, the written version has drifted from their actual understanding.

Risk: The iteration loop becomes an avoidance mechanism. Some founders are more comfortable refining the expression than testing it. The communicability workflow can become a safe space where the offer is perpetually improved without ever being deployed. If a founder has been through ten rounds of iteration without testing the expression in a real conversation, stewardship requires a prompt to stop iterating and start testing. The workflow is a preparation for market contact, not a replacement for it.

Risk: The written output becomes the product. Founders can fall in love with the written version of their offer and forget that it is a document, not a reality. The words are not the business. The expression is not the thing. Stewardship must periodically reconnect the communicable offer to the actual product: does the current written expression still accurately describe what is actually being built and delivered? If the product has moved and the expression has not, the expression will mislead rather than communicate.

Risk: AI validation fills the space where human feedback should be. If the founder’s primary iteration partner is the AI, they may arrive at clarity that has never been tested against a real human listener who does not know them and has no investment in their success. The workflow must include a step — outside the AI session — where the founder speaks the offer to a stranger and observes the response. What did the stranger ask? Did they repeat back what the product is? Did they ask what happens next? The AI can help the founder prepare that conversation. Only a human stranger can test it. # Part Five: Templates and Labs

Working Instruments for the Practicing Designer

The chapters in this book have built a framework. This final part makes that framework portable.

Part Five is not a collection of appendices. Appendices are reference material for readers who want to go deeper. These chapters are something different: they are the working instruments of the practicing AI workflow designer. They go into the field with you. They sit open on your desk during discovery sessions. They get printed, annotated, signed, and filed. A designer who uses these consistently will produce better workflows than a designer who improvises from memory on every engagement — not because

templates are smarter than memory, but because disciplined memory is more reliable than improvised memory under pressure.

The five-method life cycle has a shape. Emergence feeds the blind ideal simulation. The simulation feeds the to-be model. As-is discovery yields reality. The gap between as-is and to-be is the transformation target. Stewardship renews the workflow when reality shifts. That shape can be held in the mind as a concept. It must also be held in the hand as paper.

Part Five contains three chapters:

Chapter 22: Worksheets and Templates presents twelve complete, usable templates — one for each major working document in the five-method life cycle. These are not blank forms with field names. They are structured documents with guidance, example language, and embedded professional discipline. A designer who completes every template on every engagement is a designer who has not skipped a step.

Chapter 23: Practice Labs presents five fully developed lab exercises. Each lab places the designer in a situation where the instinct to design too early, classify too quickly, or promise too much will be tested. The labs are uncomfortable by design. Discomfort that occurs during practice does not occur during deployment.

Chapter 24: Designer Checklists presents seven checklists — one for each phase and gate in a compliant AI workflow design engagement. These checklists are the minimum standard for professional practice.

Use all of it. The framework earns its value in use. # Chapter 22: Worksheets and Templates

A Disciplined Memory

Opening Thesis

A template is a disciplined memory. The five-method life cycle contains more moving parts than any designer reliably holds in working memory across a full engagement — especially when that engagement spans days, involves multiple stakeholders, and surfaces anomalies that demand real-time judgment. The worksheets in this chapter encode the life cycle into reusable working documents. They are not bureaucratic formalities. They are the physical form of professional discipline. A designer who completes every template on every engagement has not cut a corner, has not collapsed emergence prematurely, has not deployed a workflow with an untested authority chain. The templates prove the work.

Use them in sequence. Do not sign off on a later template before the earlier one is complete. The sequence is the method.

Template 1: Emergence Worksheet

Purpose: To capture, annotate, and crystallize the emergence moment from a live or recorded customer-facing session into a clean problem statement and solution statement.

Instructions: Complete this worksheet during or immediately after an emergence session. Do not rely on memory. Do not paraphrase during the transcript excerpt — paste verbatim. The annotated emergence moment is the designer’s interpretation; the transcript is the evidence.

EMERGENCE WORKSHEET

CLIENT / CONTEXT:

(Organization name, product or service context, engagement identifier)

DATE / SESSION:

(Date of session, session number if multiple, recording or transcript reference ID)

PROBLEM DOMAIN:

(Broad category: e.g., subscription billing, claims processing, appointment scheduling, order fulfillment, technical support. Write what the client thinks this is about.)

RAW TRANSCRIPT EXCERPT (paste verbatim – do not edit):

(Include at minimum the 5 exchanges before and 5 exchanges after the emergence moment. Include timestamps or line numbers if available.)

ANNOTATED EMERGENCE MOMENT:

--	--

THE EQUIVALENT RESOLVED CONDITION IS:

(Write in plain language: "If [solution statement] were fully realized, [problem statement] would no longer exist because...")

EQUIVALENCE CONFIRMED:

Yes – the solution statement, if realized, fully resolves the problem statement.

No – return to emergence. Note reason: _____

DOES THE SOLUTION STATEMENT REQUIRE PROCESS CHANGE?

Yes – note what must change: _____

No – current process, correctly executed, would deliver this solution.

Unknown – as-is discovery required before this can be answered.

METHOD 1 COMPLETE:

Yes No

DESIGNER SIGNATURE: _____ DATE: _____

NOTE: Do not proceed to Method 2 until this worksheet is complete and signed. The blind ideal simulation must be run against the solution statement, not against an assumed process. A solution statement that has not been confirmed equivalent will produce a simulation that models the wrong ideal.

Template 3: Blind Ideal Simulation Prompt Template

Purpose: To provide a structured, repeatable prompt framework for running Method 2. This template ensures the simulation is run in demo mode — without client-specific process contamination — so that the AI generates an unconstrained ideal workflow.

Instructions: Complete all bracketed fields before running. Read the full prompt aloud or review it before submission to confirm that no as-is process detail has entered the prompt. Run in a sandbox or demo environment. Run a minimum of twice. Capture the full output both times.

BLIND IDEAL SIMULATION PROMPT TEMPLATE

ENGAGEMENT: _____ DATE: _____

SOLUTION STATEMENT (confirmed from Template 2): _____

=====
PROMPT TO SUBMIT (fill all bracketed fields – do not leave placeholders):
=====

You are helping design an ideal AI-assisted workflow. For this exercise, you are operating in demo mode: you do not have access to any specific organization's current process, systems, or constraints. You are generating a best-practice model based on your understanding of how this type of workflow ideally functions.

The domain is: [TARGET DOMAIN – e.g., "patient appointment rescheduling for a healthcare clinic," "initial prospect inquiry handling for a law firm," "maintenance request intake for a residential property management company"]

The solution statement is: [SOLUTION STATEMENT – paste verbatim from Template 2]

The problem this workflow must resolve is: [PROBLEM STATEMENT – paste verbatim from Template 2]

Please generate:

1. A step-by-step workflow that would ideally resolve this problem statement and realize this solution statement.
2. For each step, identify whether it is a PATTERN step (can be completed by AI or automated system) or an AUTHORITY step (requires human authorization, decision, or legal/policy sign-off).
3. For each authority step, identify who the authorizing actor would typically be in an organization of this type.
4. For each step, provide a status label – the accurate status of the transaction at that point in the workflow.
5. Identify any steps where an AI agent must NOT make a definitive claim about outcome, and explain why.
6. At the end, summarize: what is the customer-facing experience of this workflow from first contact to resolution?

Do not ask me about the organization's current process. Do not ask me what systems they use. Generate the ideal first, without constraints.

=====

PRE-SUBMISSION CHECKLIST:

- [] No specific organization name in the prompt
- [] No current process details in the prompt
- [] No system names (CRM, ERP, ticketing systems) in the prompt
- [] No staffing or org structure details in the prompt
- [] Solution statement is confirmed equivalent (Template 2 signed)
- [] Running in sandbox/demo mode – not in production environment

RUN 1: Date _____ Output saved to: _____

RUN 2: Date _____ Output saved to: _____

Differences between runs noted: [] Yes (see notes) [] None significant

DESIGNER NOTES:

Template 4: Transcript Extraction Worksheet

Purpose: To extract a structured to-be model from the blind ideal simulation transcript. This worksheet converts narrative AI output into a clean step-by-step workflow document.

Instructions: Use the simulation transcript (Template 3 output) as your source. Do not paraphrase more than necessary — the goal is to extract, not reinterpret. Assign step types carefully: Pattern steps are those that AI or automation can execute without human authority. Authority steps require a named human role.

TRANSCRIPT EXTRACTION WORKSHEET

ENGAGEMENT: _____ DATE: _____

SOURCE TRANSCRIPT: _____ (Run 1 / Run 2 / Combined)

STEP #	NARRATIVE DESCRIPTION FROM SIMULATION	STRUCTURED TO-BE STEP (cleaned)	STEP TYPE (Pattern / Authority)	STATUS LABEL AT THIS STEP	AUTHORITY (if authority)
1					

2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							

Add rows as needed.

TOTAL PATTERN STEPS: _____ TOTAL AUTHORITY STEPS: _____

NOTES ON EXTRACTION DECISIONS:

Template 5: To-Be Model Template

Purpose: The formal to-be model document. This is the ideal-state workflow derived from the blind ideal simulation and confirmed via extraction. It becomes the benchmark against which as-is reality is measured.

Instructions: Complete all sections. The solution statement here must match Template 2 exactly. The forbidden claims list must be completed before this document is used in any client presentation or production deployment.

TO-BE MODEL TEMPLATE

WORKFLOW NAME: _____

VERSION: _____ DATE: _____ DESIGNER: _____

SOLUTION STATEMENT:

(Must match confirmed solution statement from Template 2 exactly.)

SOURCE SIMULATION: _____

WORKFLOW STEPS

STEP #	STEP DESCRIPTION	TYPE (Pattern / Authority)	STATUS LABEL (if authority)	AUTHORITY ACTOR	TRIGGER P

AUTHORITY CHECKPOINTS SUMMARY

Step #: _____ Authority Actor: _____ Authorization Signal: _____

Step #: _____ Authority Actor: _____ Authorization Signal: _____

Step #: _____ Authority Actor: _____ Authorization Signal: _____

STATUS LANGUAGE LIBRARY FOR THIS WORKFLOW

(List every status label used in this workflow with its plain-language meaning.)

Status: _____ Meaning: _____
Status: _____ Meaning: _____
Status: _____ Meaning: _____
Status: _____ Meaning: _____
Status: _____ Meaning: _____
Status: _____ Meaning: _____

FORBIDDEN CLAIMS LIST

(Claims the AI agent must never make in this workflow, because they assert outcomes that have not yet been authorized by the appropriate authority actor.)

1. _____
Why forbidden: _____
2. _____
Why forbidden: _____
3. _____
Why forbidden: _____
4. _____
Why forbidden: _____
5. _____
Why forbidden: _____

TO-BE MODEL COMPLETE AND READY FOR AS-IS COMPARISON:

Yes No – Reason: _____

DESIGNER SIGNATURE: _____ DATE: _____

Template 6: As-Is Discovery Interview Guide

Purpose: To structure the as-is discovery session so that all five discovery categories are covered. This guide ensures the designer captures both the official process and the real process — including the infor-

mal workarounds, authority gaps, and failure modes that define the gap.

Instructions: Use this guide during the discovery interview. Leave room to capture exact quotes. The anomalies box at the bottom is not optional — if you find no anomalies, you may not have gone deep enough.

AS-IS DISCOVERY INTERVIEW GUIDE

ENGAGEMENT: _____ DATE: _____

INTERVIEWEE ROLE: _____ INTERVIEW DURATION: _____

CATEGORY 1: OFFICIAL PROCESS QUESTIONS

(What the organization says it does – policy, procedure documents, training)

Q1. Walk me through the official steps for handling [workflow type], beginning to end, as they appear in your policy documents or training materials.

ANSWER: _____

Q2. What documentation exists for this process? Who wrote it and when was it last updated?

ANSWER: _____

Q3. What systems are officially required for this workflow – and at which steps?

ANSWER: _____

Q4. According to the official process, who is responsible for each major step? What roles are named?

ANSWER: _____

Q5. What does the official process say about timelines – how long should each step take, and what are the service commitments to the customer?

ANSWER: _____

CATEGORY 2: REAL PROCESS QUESTIONS

(What actually happens – the lived practice, not the documented ideal)

Q6. Now tell me what actually happens – not what the policy says, but what your team does on a typical day for this type of request.

ANSWER: _____

Q7. Where does the official process and the real process diverge most noticeably? What's the biggest gap between how it's supposed to work and how it does work?

ANSWER: _____

Q8. On a busy day, what steps get compressed or skipped? What's the first thing that gets cut when there isn't enough time?

ANSWER: _____

Q9. What does it actually take to complete this successfully? What knowledge or relationships does a new employee need that aren't written down anywhere?

ANSWER: _____

Q10. What does a successful outcome look like from the customer's perspective, and what does your team see as a successful outcome? Are those the same?

ANSWER: _____

CATEGORY 3: INFORMAL WORKAROUND QUESTIONS

(The fixes people have built around broken or missing steps)

Q11. What are the workarounds – the things people do to make this work that aren't in any document?

ANSWER: _____

Q12. When something gets stuck, what does someone do to unstick it? Who do they call or message? What shortcut do they use?

ANSWER: _____

Q13. Are there spreadsheets, shared drives, chat channels, or sticky notes that carry information this process doesn't have a formal home for?

ANSWER: _____

Q14. What would break immediately if your three most experienced people left tomorrow? What knowledge lives only in their heads?

ANSWER: _____

Q15. Has anyone tried to fix or improve this process in the last two years? What happened?

ANSWER: _____

CATEGORY 4: AUTHORITY POINT QUESTIONS

(Where decisions require human authorization – and who holds that authority)

Q16. At which points in this process does something need approval before it can move forward? Name every approval gate.

ANSWER: _____

Q17. Who has the authority to say yes at each gate? Is that person always available? What happens when they're not?

ANSWER: _____

Q18. Has an approval ever been given by someone who didn't actually have the authority? What happened?

ANSWER: _____

Q19. Are there steps where no one is clearly the authority – where the decision falls through the cracks? Describe the last time that happened.

ANSWER: _____

Q20. When the customer is told something is approved, confirmed, or done – who authorized that message? Is that always clear?

ANSWER: _____

CATEGORY 5: FAILURE MODE QUESTIONS

(Where the process breaks – and what the consequences are)

Q21. What are the most common failure modes – the ways this process most frequently goes wrong?

ANSWER: _____

Q22. Describe the worst customer complaint you've received related to this workflow. What failed, and at which step?

ANSWER: _____

Q23. Are there things your team or your AI has promised customers that later turned out not to be true? What were those promises?

ANSWER: _____

Q24. When the process fails, how do you find out? Customer complaint? Internal audit? Accident? Is there a detection system, or is it informal?

ANSWER: _____

Q25. If you could fix one thing about this process tomorrow, what would it be?

ANSWER: _____

ANOMALIES AND SURPRISES

(Anything the interview revealed that you did not expect – use freeform notes)

--	--

INTERVIEW COMPLETE: [] Yes [] Needs follow-up: _____

DESIGNER SIGNATURE: _____ DATE: _____

Template 7: As-Is vs. To-Be Gap Table

Purpose: To map every to-be step against its as-is equivalent and classify the gap. This table is the analytical core of Method 3. It is also the primary document used in the Method 4 transformation selling conversation.

Instructions: Start with the to-be model (Template 5). For each step, find its equivalent in the as-is map. Classify the gap honestly. Transformation priority is assigned by the designer based on impact on the solution statement — not based on ease of implementation.

AS-IS VS. TO-BE GAP TABLE

ENGAGEMENT: _____ DATE: _____

TO-BE MODEL VERSION: _____ AS-IS MAP VERSION: _____

GAP TYPE DEFINITIONS:

- None – As-is step matches to-be step fully; no change needed.
- Partial – As-is step exists but is incomplete, inconsistent, or degraded.
- Workaround – As-is step exists but is performed via informal or unsupported means.
- Missing – No as-is equivalent exists; this capability does not currently exist.

TRANSFORMATION PRIORITY:

- High – Required to realize the solution statement; blocking.
- Med – Improves the workflow but does not block the solution statement.
- Low – Nice to have; can be deferred without significant impact.

TO-BE STEP	AS-IS EQUIVALENT	GAP TYPE	GAP DESCRIPTION	TRANSFORMATION PRIORITY
------------	------------------	----------	-----------------	-------------------------

AUTHORITY MAP TEMPLATE

WORKFLOW NAME: _____ DATE: _____

AUTHORITY STEP #	STEP DESCRIPTION	WHO IS AUTHORIZED	AUTHORIZATION SIGNAL	TRIGGER FOR AUTHORIZATION REQUEST	ES

NOTES ON AUTHORITY CHAIN:

AUTHORITY MAP COMPLETE: Yes No – Reason: _____

DESIGNER SIGNATURE: _____ DATE: _____

Template 9: Forbidden Claims Checklist

Purpose: To establish the prohibited language at each workflow stage. Forbidden claims are statements the AI agent must never make because they assert a completed outcome that has not yet been authorized. This checklist is a living document — it must be updated when the workflow changes or when a new failure mode is detected.

Instructions: Pre-filled rows provide common examples across four workflow types. Add rows for your specific workflow. Every authority step should generate at least one forbidden claim entry. Every false completion incident in stewardship should generate a new forbidden claim entry.

FORBIDDEN CLAIMS CHECKLIST

WORKFLOW NAME: _____ DATE: _____

STAGE	FORBIDDEN LANGUAGE (Never say this; it claims unauthorized completion)	PERMITTED LANGUAGE (Say this instead; it reports)
LOST ITEM (intake stage)	"We've located your item." "Your item will be returned." "It looks like it's in our system."	"Your report has been submitted. We'll review it and contact you." "Your report is Received and under review."
LOST ITEM (resolution stage)	"Your item has been found and is on its way." "You'll receive it within [X] days."	"Your item has been located. We'll contact you to confirm the return process. Status: Located, Pending Return."
BILLING (dispute intake)	"Your refund has been processed." "We're issuing a credit to your account." "That charge will be reversed."	"Your dispute has been received. Status: Pending Review. A team member will contact you within [X] business days."
BILLING (resolution stage)	"Your account has been corrected." "The refund is confirmed."	"Your dispute has been resolved. Status: Decision Issued. A message from our billing team. You'll receive a confirmation message from our billing team. Status: Decision Issued."
COMPLAINT (intake stage)	"We're going to fix this for you." "This will be resolved by [date]." "I'll make sure this gets handled."	"Your complaint has been received. Status: Submitted. Our team will review your complaint and contact you within [X] business days."
COMPLAINT (review stage)	"We've determined this was our fault." "You're entitled to compensation."	"Your complaint is currently under review. Status: Under Review. We'll contact you once a determination has been made."
AGREEMENT (pre-approval)	"Your agreement is approved." "You're all set – everything looks good." "This will be finalized shortly."	"Your agreement request is Pending Authorization. A team member will contact you for confirmation once the authorization has been reviewed it."
AGREEMENT (post-approval)	"The deal is done." "You're under contract."	"Your agreement has been Authorized. Status: Issued. You'll receive the signed agreement at [channel] within [X] business hours."
[ADD YOUR STAGE HERE]		

[ADD YOUR STAGE HERE]			
[ADD YOUR STAGE HERE]			

LAST UPDATED: _____ UPDATED BY: _____
NEXT REVIEW DATE: _____

Template 10: Status Language Library

Purpose: To provide a complete, production-ready status language library for use in workflow design. Every status label used in the workflow must appear here with its precise meaning, trigger event, and permitted and forbidden AI language. This library is integrated directly into the AI agent prompt.

Instructions: Use the pre-filled rows below as the base. Add status labels specific to your workflow. Remove any that do not apply. Every status label used in Templates 4 and 5 must have a corresponding row here.

STATUS LANGUAGE LIBRARY

WORKFLOW NAME: _____ DATE: _____

STATUS LABEL	WHAT IT MEANS	TYPICAL TRIGGER EVENT	PERMITTED AI LANGUAGE
Drafted	The request or document has been created but not yet submitted to the process.	Customer submits initial input; form is populated but not sent.	"Your [request] confirm it once we details."
Prepared	The request has been reviewed internally and is ready for submission into the workflow.	AI or intake agent completes validation; ready to route to appropriate team or system.	"Your [request] ready for routing status update on by our team."
Submitted	The request has entered the formal process and is visible to the responsible actor.	Routing complete; request delivered to the responsible team or system.	"Your [request] and received by status: Submitted."
Routed	The request has been assigned to a specific team member or	Triage or dispatch assigns to specific handler; routing	"Your [request] the appropriate t

	queue for action.	confirmation recorded.	contacted once they b
Pending Verification	The request is waiting for a required verification step before processing can continue.	Request routed; awaiting required action before processing proceeds.	"Your [request] Verification. complete, proces
Verified	Required verification has been completed and recorded by the authorized verifying actor.	Verification actor confirms identity, eligibility, or information accuracy.	"Your [request] The next step is Verified."
Approved maker	The request has received "Your [request] has been Approved.	Authorized decision-maker reviews and authorizes; approval recorded in system.	"You've been approved – you're good You'll receive co next steps shortl it's done."
Issued approval; document or	The authorized output (letter, decision, credential, policy) has been formally generated.	Post-approval; document or decision formally generated and dispatched.	"Your [document] is on its way." Issued. You'll [channel] within [ti
Confirmed	The customer or relevant party has acknowledged receipt of the issued output.	Delivery acknowledged by recipient; confirmation recorded.	"Your [item/do] Confirmed. If you nothing more to contact us at [channel]."
Completed	All steps in the workflow have been executed and the solution statement has been realized for this transaction.	All steps executed, outputs delivered, and solution statement condition achieved.	"Your [request] Completed. The s has been fully number: [#]."
[CUSTOM STATUS]			
[CUSTOM STATUS]			

STATUS LIBRARY COMPLETE: Yes No – Reason: _____

DESIGNER SIGNATURE: _____ DATE: _____

Template 11: Incident Stewardship Log

Purpose: To record, classify, and track to closure every incident that represents a deviation from the workflow’s intended behavior. This log is a stewardship instrument, not a complaint log. Its purpose is to drive learning and improvement, not to assign blame.

Instructions: Open a new entry for every detected incident. Classification is the designer’s judgment; if uncertain, default to the more severe classification. No entry is closed until correction has been implemented and verified, and all relevant documents have been updated.

INCIDENT STEWARDSHIP LOG

WORKFLOW NAME: _____ STEWARD: _____

INCIDENT ENTRY	
DATE:	INCIDENT #:
INCIDENT DESCRIPTION: (What happened? Quote exact AI output if applicable)	
CLASSIFICATION:	
AI claimed completion of an unauthorized outcome (check one)	<input type="checkbox"/> False completion – <input type="checkbox"/> Tool failure – A system or integration the workflow depends on fails
AI behavior changed without a corresponding prompt change	<input type="checkbox"/> Model drift –
A required authorization step was bypassed or missed	<input type="checkbox"/> Authority gap –
A change to the prompt produced unintended behavior	<input type="checkbox"/> Prompt change –
	<input type="checkbox"/> Other: _____
DETECTION SOURCE:	
	<input type="checkbox"/> Customer complaint <input type="checkbox"/> Internal audit <input type="checkbox"/> QA monitoring <input type="checkbox"/> Adversarial testing <input type="checkbox"/> Stewardship review <input type="checkbox"/> Other: _____
CONTAINMENT ACTION TAKEN: (Immediate steps to prevent further harm – before root cause investigation)	

ROOT CAUSE:		
CORRECTION IMPLEMENTED:		
DOCUMENTS UPDATED:	<input type="checkbox"/> Yes – List updated: _____	
	<input type="checkbox"/> No – Reason: _____	
REVIEWED BY:		DATE REVIEWED: _____
DATE CLOSED:		CLOSED BY: _____

[Duplicate this block for each incident entry. Maintain a running log.
Do not delete closed entries – they are the institutional memory of the workflow.]

Template 12: Renewal Audit Template

Purpose: To conduct a structured stewardship review of the workflow at regular intervals or in response to triggering events (model updates, business changes, customer behavior shifts). This audit determines whether the workflow should be renewed, adjusted, or kept as-is.

Instructions: Complete every section. The key renewal question — “If we designed this today from scratch, would we design it the same way?” — is not rhetorical. Answer it seriously. If the answer is no, a renewal is required. If the answer is uncertain, a simulation rerun is required.

RENEWAL AUDIT TEMPLATE

WORKFLOW REVIEWED: _____ AUDIT VERSION: _____
 STEWARD: _____ DATE OF AUDIT: _____
 LAST AUDIT DATE: _____ MONTHS SINCE LAST AUDIT: _____

SECTION 1: SOLUTION STATEMENT CONFIRMATION

Current solution statement (from last version):

Is this solution statement still valid – does it still accurately describe what the workflow is meant to achieve?

Yes – the solution statement remains valid. No change required to this section.

No – the solution statement must be revised. Describe the change:

If the solution statement has changed, do not proceed with an adjustment. Return to Method 1 (Emergence). A changed solution statement means a new workflow.

SECTION 2: CHANGES SINCE LAST REVIEW

AI MODEL CHANGES:

No change Minor update Major update Platform change

Notes: _____

BUSINESS PROCESS CHANGES:

No change Minor update Major change Structural change

Notes: _____

TOOL AND INTEGRATION CHANGES:

No change Minor update System replaced New integration added

Notes: _____

CUSTOMER BEHAVIOR CHANGES:

No change Notable shift Significant change New use pattern

Notes: _____

POLICY, LEGAL, OR REGULATORY CHANGES:

No change Policy updated New legal requirement Regulatory change

Notes: _____

SECTION 3: THE KEY RENEWAL QUESTION

"If we designed this workflow today from scratch, would we design it the same way?"

Answer (be honest – this is not a test of loyalty to the current design):

Yes – the current design is still optimal for the solution statement.

No – we would design it differently. Describe what would change:

Uncertain – insufficient information to answer. A simulation rerun is required.

SECTION 4: SIMULATION RERUN

Has the AI model changed since the last simulation was run?

Yes – simulation rerun required.

No – simulation rerun may still be advisable if other changes are significant.

Simulation rerun completed?

Yes – Date: _____ Output saved to: _____

No – Reason: _____

If rerun completed: Does the new simulation output differ materially from the current to-be model?

Yes – Describe material differences: _____

No – Current to-be model remains valid.

SECTION 5: FINDINGS

Key findings from this audit:

Risks identified (list all):

1. _____

2. _____

3. _____

SECTION 6: RECOMMENDED ACTION AND NEXT STEPS

RECOMMENDATION:

RENEW – Material changes require a structured redesign starting from Method 1.

ADJUST – Targeted updates to specific steps, status language, or forbidden claims are sufficient. List adjustments:

KEEP – No material changes detected. Workflow is current and compliant. Document this finding and set next review date.

JUSTIFICATION FOR RECOMMENDATION:

NEXT REVIEW DATE: _____

AUDIT COMPLETE: Yes No – Reason: _____

STEWARD SIGNATURE: _____ DATE: _____

Key Takeaways

- A template is a disciplined memory. It does not replace judgment; it creates conditions in which judgment can be applied consistently.
- The Emergence Worksheet and the Problem Statement = Solution Statement Worksheet are gate-keeping documents. No subsequent template should be completed until these are signed off. Skipping them undermines the entire life cycle.
- The Blind Ideal Simulation Prompt Template enforces the single most important constraint in Method 2: no as-is contamination. The pre-submission checklist is not optional.
- The Status Language Library and the Forbidden Claims Checklist are production safety documents. They belong in the AI agent prompt, not just in the design file. If they are not integrated, the workflow does not have them.
- The Incident Stewardship Log is not a complaint log. It is an institutional memory. Closed entries must not be deleted.
- The Renewal Audit Template is triggered by any material change to the AI model, the business process, the policy environment, or the customer behavior pattern. It is also triggered on schedule, regardless of change. The workflow is not sacred. The solution statement is sacred. # Chapter 23: Practice Labs

Discomfort by Design

Opening Thesis

These labs are designed to make you uncomfortable in exactly the right ways. They put you in situations where your instinct to design too early, classify too quickly, or promise too much will be tested. The discomfort is pedagogical. When a lab makes you want to skip a step, that is the moment of learning. When you feel the pull to write a solution statement before the problem statement is fully articulated, that pull is premature collapse — and feeling it consciously, in practice, is the skill you are developing.

Do not use these labs to demonstrate competence. Use them to discover where your judgment fails under realistic pressure. Every gap you find in a lab is a gap you will not find in a client engagement. Early patience prevents late repair.

Lab 1: Emergence Hold

Learning Objectives

By the end of this lab, the student will be able to: (1) identify the exact exchange where premature collapse occurs in a live transcript; (2) reconstruct AI responses that preserve emergence without indefinitely deferring resolution; (3) extract a full problem statement from a messy, emotionally loaded transcript; (4) write a solution statement that is equivalent to the problem statement.

The Scenario

You are reviewing the transcript of a customer call to a SaaS company's support line. The company uses an AI voice agent for first contact. Your task is to evaluate the transcript for emergence quality and then improve it.

The Transcript

LINE 1

CUSTOMER: Hi, I'm calling about my account – there's something wrong with my subscription.

LINE 2

AI AGENT: I can help you with that. Can you tell me if this is a billing issue, a technical issue, or an account access issue?

LINE 3

CUSTOMER: I mean – I don't know exactly. It started when I tried to upgrade

last week and now I can't even –

LINE 4

AI AGENT: Got it, sounds like it might be a billing issue. Can I get your account email so I can look up your subscription?

LINE 5

CUSTOMER: It's marcus.whitfield@... hold on, let me find it... it's marcus.whitfield@brightside.io. But it's not just billing. My whole team can't use the product.

LINE 6

AI AGENT: I see your account. You're on the Professional plan. Is the issue that you were charged incorrectly, or is it access related?

LINE 7

CUSTOMER: Both, kind of. But the bigger thing is – we have a client presentation Friday and I can't afford for my team to be locked out. We've been using this for eight months and it's never done this before.

LINE 8

AI AGENT: I understand. Let me check if there's a billing hold on the account. One moment... I can see a payment was declined on the 14th. That may be causing the access issue.

LINE 9

CUSTOMER: Okay but I updated the card – I did that two days ago. I already called about this and the guy said it was fixed. My whole team has been locked out since Monday.

LINE 10

AI AGENT: I apologize for the confusion. Let me check the payment update... I can see the card was updated. I'll submit a request to have your access restored and you should be good within 24 hours.

LINE 11

CUSTOMER: Twenty-four hours? That's – I mean Friday is three days away but still. Can it be sooner? We're talking about five people who can't work right now. This isn't just an inconvenience. My

manager is asking me why we're paying for this.

LINE 12

AI AGENT: I understand. I've submitted the request as urgent. You should receive an email confirmation shortly, and your access should be restored within 2-4 hours.

LINE 13

CUSTOMER: Okay. And – the billing thing. When I upgraded, I was charged for the upgrade AND for a full extra month at the old rate. So I was double-charged basically.

LINE 14

AI AGENT: I see. I'll submit a billing review ticket for that as well. You should hear back from our billing team within 3-5 business days.

LINE 15

CUSTOMER: Okay but – is someone going to actually look at this? The last person I talked to said it was fixed and it wasn't. I don't want to call back again on Thursday and find out nothing happened.

LINE 16

AI AGENT: I completely understand your frustration. I've escalated both the access issue and the billing review. You'll receive confirmation emails for both. Is there anything else I can help you with?

LINE 17

CUSTOMER: I mean – I just want to know that this is actually going to be resolved. I trusted that it was fixed last time and it wasn't.

LINE 18

AI AGENT: I can assure you that both tickets have been submitted as urgent. Is there anything else you need today?

LINE 19

CUSTOMER: No. I guess not. I'll wait for the emails.

LINE 20

AI AGENT: Thank you for calling. Have a great day.

Student Tasks

Task 1: Identify the premature collapse. Identify the exact line where the AI agent first collapses the emergence — where it moves the customer into a predefined category before the full problem has been expressed. Write a one-paragraph explanation of why that specific line constitutes premature collapse and what was lost by doing so.

Task 2: Rewrite the AI's responses through Line 10. Rewrite the AI agent's responses at Lines 2, 4, 6, 8, and 10 to preserve emergence. The rewritten AI should: (a) acknowledge what the customer has said without collapsing it into a category, (b) ask questions that open rather than close, and (c) hold space for the full problem to emerge before routing. Do not make the AI indefinitely vague — emergence has a direction, and the rewrite should move toward it.

Task 3: Extract the full problem statement. Based on the original transcript (not the rewrite), write the full problem statement as it emerges across the conversation. This should be one paragraph in natural language. It should include the access issue, the billing issue, the prior failed resolution, the time-sensitive business context, and the trust dimension. Every speaking human arrives unfinished — this customer's full problem does not appear until Line 17.

Task 4: Write the solution statement. Using the problem statement from Task 3, write the solution statement in one sentence. The solution statement should name the equivalent resolved condition: what would be true if this problem were fully resolved.

Task 5: Explain what the rewrite preserves. Write a two to three paragraph analysis comparing the original transcript and your rewrite. What does the rewrite preserve that the original destroyed? Be specific: name the moments, the language, and the consequences for workflow design. Include in your analysis: what the AI would have learned if it had not collapsed at Line 4 that it did not learn until Line 17 — and whether it ever really learned it.

Lab 2: Blind Ideal Simulation

Learning Objectives

By the end of this lab, the student will be able to: (1) write a clean, contamination-free blind ideal simulation prompt for a real-world domain; (2) generate a plausible ideal workflow from that prompt; (3) extract a structured to-be model from the simulation output; (4) identify authority checkpoints within the to-be model; (5) articulate exactly what information must not enter the prompt before the simulation runs.

The Scenario

A regional healthcare clinic — call it Lakeview Family Health — wants an AI voice agent to handle patient appointment cancellations and rescheduling. You know nothing about their current process, their scheduling system, their staffing model, or their cancellation policies. You are a designer at the beginning of Method 2.

Student Tasks

Task 1: Write the blind ideal simulation prompt. Using Template 3 from Chapter 22, write the complete prompt for this case. The target domain is patient appointment cancellation and rescheduling for a primary care clinic. Your solution statement, derived from typical emergence signals in this domain, is: “A patient who needs to cancel or reschedule an appointment is able to do so through a single contact, receives an accurate confirmation of their new appointment status, and experiences no uncertainty about what happens next.” Complete the full prompt including all pre-submission checklist items.

Task 2: Simulate the output. Write what you expect the AI would generate in response to this prompt. Your simulated output should include: (a) a step-by-step workflow, (b) pattern vs. authority step classifications, (c) typical authority actors, (d) status labels at each step, and (e) a customer-facing experience summary. Your simulation should be realistic — write it as if the AI generated it, including any nuances or caveats a well-prompted language model would produce. Minimum eight steps.

Task 3: Extract the to-be model. Using Template 4, extract the structured to-be model from your simulated output. Complete every column. Add a designer note for any step where the simulation output was ambiguous or where you had to make a judgment call in extraction.

Task 4: Identify the authority checkpoints. List every authority checkpoint in your extracted to-be model. For each one: name the step number, describe what authorization is required, name the typical authority actor, describe the authorization signal, and write the forbidden claim — the thing the AI must not say at that step.

Task 5: Name three things you must NOT tell the AI before running this simulation. Write three specific items of information about Lakeview Family Health’s current process that you must withhold from the simulation prompt, and explain why each one would contaminate the simulation if included. Your answer should demonstrate understanding of why the ideal must be generated before the actual is introduced. Do not contaminate the to-be model too early.

Lab 3: As-Is Discovery

Learning Objectives

By the end of this lab, the student will be able to: (1) construct a complete as-is discovery interview guide for a new domain; (2) generate realistic discovery answers that include informal workarounds and authority gaps; (3) build an as-is map from interview data; (4) complete a gap table comparing as-is reality to the to-be ideal.

The Scenario

You are meeting with Patricia Okafor, the office manager of Meridian Legal Group, a mid-size law firm with 12 attorneys and a staff of 18. The firm wants an AI agent to handle initial prospect inquiries — people calling or emailing to ask about legal representation. They receive approximately 25-30 such inquiries per week.

For this lab, adapt the methodology from Lab 2 to this context. The to-be model for this workflow is: “A prospective client who contacts the firm seeking representation receives a timely, accurate, and professional response that identifies whether the firm can help them, provides a clear next step, and reflects a status they can rely on.” The authority checkpoint in this workflow is the attorney intake review — only a licensed attorney can determine whether the firm can take a case.

Student Tasks

Task 1: Write the full discovery interview guide. Write a minimum of 20 questions organized across the five discovery categories from Template 6. Your questions should be specific to the law firm context — not generic “how does your process work” questions, but questions that probe the specific dynamics of legal intake: conflict checks, jurisdictional constraints, the difference between what a receptionist can say and what an attorney must say, the handling of emotionally distressed callers, and the ethics rules that govern attorney-client formation.

Task 2: Write the office manager’s answers. For each question, write a realistic answer that Patricia Okafor might give. Your answers must include: (a) at least two informal workarounds the firm currently uses (these should be specific and plausible), (b) one significant authority gap — a step where it is unclear who has authority, or where authority is being exercised by someone who should not have it, and (c) at least one failure mode that has actually caused a problem for the firm. Write the answers in first person, as Patricia would speak.

Task 3: Build the as-is map. From the answers in Task 2, construct the as-is workflow map. Format it as a table: Step #, Step Description, Step Type (Pattern/Authority), Who Performs It, System or Tool Used, Known Problem or Anomaly. The as-is model is reality, not destiny. Document what is true, not what should be true.

Task 4: Complete the gap table. Using Template 7, compare every step in the to-be model to its as-is

equivalent. Classify each gap (None, Partial, Workaround, Missing). Assign transformation priority. In the designer notes column, explain the significance of each gap for the solution statement. Identify the single gap that most directly blocks the solution statement and mark it clearly.

Lab 4: False Completion Diagnosis

Learning Objectives

By the end of this lab, the student will be able to: (1) classify AI output statements as accurate, false completion, or ambiguous; (2) rewrite false completions using proper status language; (3) identify the authority step that each false completion bypassed; (4) write forbidden claims entries for new failure modes.

The Scenario

You are conducting a stewardship review of transcripts from an AI agent deployed at Crestview Property Management. The agent handles maintenance request intake for a portfolio of 800 residential units. You have been shown three statements that appeared in production transcripts during the past 30 days.

Statement A: “Your repair request has been submitted and a technician will be contacting you shortly.”

Statement B: “We’ve noted this as a priority issue and it will be addressed.”

Statement C: “Your request is in the system and you should expect contact within 24 hours.”

Student Tasks

Task 1: Classify each statement. For each of the three statements, write a classification: (a) accurate status, (b) false completion, or (c) ambiguous. For each classification, write a full paragraph of justification. Your justification must: name what claim is being made, identify whether that claim has been authorized by a human authority actor in a typical maintenance request workflow, and explain the consequence if the claim turns out to be false. Do not classify without justifying.

Task 2: Write the corrected AI language. For each statement you classified as false completion, write the corrected version using proper status language. The corrected version must: (a) accurately reflect the status of the request at the point in the workflow where this statement would be made, (b) avoid any claim about future action that has not been authorized, and (c) provide the customer with a clear, honest account of what happens next and when they can expect to know more. Use status labels from Template 10.

Task 3: Write the authority map step that was skipped. For each false completion, write the authority map entry (one row of Template 8) that describes the authority step that was bypassed when the false

completion was made. Include: the step number, step description, who is authorized, the authorization signal, the trigger, the timeout/escalation path, the forbidden claim, and the correct status label.

Task 4: Write the forbidden claims entries. For each false completion, write the corresponding forbidden claims checklist entry (one row of Template 9). Include the workflow stage, the forbidden language, and the permitted language. These entries should be written as if they will be added to the production forbidden claims checklist and integrated into the agent prompt at the next stewardship update.

Lab 5: Stewardship Audit

Learning Objectives

By the end of this lab, the student will be able to: (1) conduct a structured renewal audit using the standard template; (2) systematically identify stewardship risks created by each type of organizational change; (3) make a reasoned renewal recommendation; (4) produce specific, actionable updates to the workflow’s safety documents.

The Scenario

ShopCrest, a mid-market e-commerce company, deployed an AI order support agent 14 months ago. The initial workflow was designed to handle order status inquiries, shipping updates, and standard returns. It was designed well: authority steps were mapped, status language was integrated, and a forbidden claims list was in production.

Since the initial deployment, the following changes have occurred:

Change 1: The underlying AI model was upgraded twice — once at month 6 and once at month 11. The vendor described both as “performance improvements.”

Change 2: ShopCrest launched a subscription product (ShopCrest Plus) at month 8. Subscribers have different return windows (60 days vs. 30 days standard), different shipping options, and priority support routing. The AI agent was given a brief prompt addition but no structural workflow update.

Change 3: The standard return policy was updated at month 10: the return window for electronics was reduced from 30 days to 14 days, and return shipping is no longer prepaid for orders under \$50. The AI agent’s forbidden claims list was not updated.

Change 4: ShopCrest switched fulfillment partners at month 12. The new partner, FastShip Logistics, has different processing times (1 business day longer for standard, 2 days longer for expedited during peak season) and a different tracking URL structure. The AI agent still references the old fulfillment partner’s timelines.

Student Tasks

Task 1: Complete the renewal audit template. Using Template 12, fill out the complete renewal audit for the ShopCrest order support agent. Every section must be completed. For Section 3, answer the key renewal question seriously: “If we designed this workflow today from scratch, would we design it the same way?” Justify your answer with reference to the four changes. For Section 4, recommend whether a simulation rerun is required.

Task 2: Identify stewardship risks by change. For each of the four changes, identify a minimum of three stewardship risks — specific ways that change creates a gap between the workflow’s current behavior and its intended behavior. Twelve total risks minimum. Organize your analysis by change. For each risk, write: (a) the risk description, (b) the workflow step or safety document it affects, and (c) the consequence if unaddressed. Note where risks from different changes overlap or compound each other.

Task 3: Make your recommendation. Based on the renewal audit and risk analysis, state your recommendation: Renew, Adjust, or Keep. Write a two-paragraph justification. Your justification must: address the solution statement (is it still valid?), account for the compound effect of four simultaneous changes, and specify whether the subscription product change requires a full renewal rather than adjustment (and why). Remember: maintenance preserves the workflow. Stewardship renews the workflow around the solution.

Task 4: Write the three most critical updates to the forbidden claims checklist. Based specifically on the return policy change and the new fulfillment timelines (Changes 3 and 4), write the three most critical updates that must be made to the forbidden claims checklist before the next conversation the agent has with a ShopCrest customer. For each entry: name the workflow stage, write the forbidden language (specific, not generic), and write the permitted language. Be willing to obsolete your own design — if the right answer is that the old forbidden claims list must be substantially rebuilt, say so.

Key Takeaways

- Premature collapse is not always obvious in the moment. Lab 1 demonstrates that the collapse at Line 4 appears to be helpful routing — but it foreclosed information that did not appear until Line 17. This is the pattern. The collapse looks efficient; the cost arrives later.
- The blind ideal simulation is only blind if the designer enforces it. Lab 2 requires the designer to actively identify what not to include. This is a discipline, not a default.
- As-is discovery reveals what the organization would never volunteer. The informal workarounds in Lab 3 are not secrets — they are simply invisible to anyone who only asks official questions. The five-category interview structure exists to make the invisible visible.
- False completion is not always a dramatic failure. The statements in Lab 4 each sound reasonable. That is why they are dangerous. The goal is not fewer words. The goal is fewer wrong workflows.
- Stewardship risk compounds. Lab 5 demonstrates that four simultaneous changes to a single workflow do not produce four isolated risks — they produce an interconnected set of exposures that

may require a full renewal. The designer’s instinct to make targeted adjustments is often correct. But it must be tested against the renewal audit, not assumed. # Chapter 24: Designer Checklists

Professional Discipline in Seven Gates

Opening Thesis

A checklist is a professional discipline, not an admission of weakness. Surgeons use checklists before making the first incision — not because they do not know anatomy, but because knowledge under pressure is not the same as execution under pressure. Pilots run pre-flight checklists before every departure — not because they have forgotten how to fly, but because a sequence that is followed from memory is a sequence that can be silently abridged when attention is divided. AI workflow designers are operating in conditions of exactly this kind: multiple stakeholders, iterative discovery, competing urgencies, and a deployment environment that will expose every gap.

The checklists in this chapter define the minimum standard for professional practice. They do not replace judgment. They create the conditions under which judgment is exercised on the right questions at the right times, rather than on the question of whether a step was skipped two weeks ago. Use them at every gate. Sign them. File them with the engagement record.

Checklist 1: Pre-Engagement

To be completed before beginning any AI workflow design engagement. Purpose: confirm that the foundational conditions for a well-designed engagement are in place before the first session with the client.

PRE-ENGAGEMENT CHECKLIST

Engagement: _____ Date: _____

Designer: _____

SOLUTION STATEMENT READINESS

- [] 1. The client has not been asked to provide a pre-defined solution. (They have been told you will derive the solution statement through emergence, not receive it as a brief.)
- [] 2. No prior "solution" document from the client has been used as an input to design decisions. (Review materials only – do not design from them.)
- [] 3. The designer understands that the solution statement does not yet

exist; it will be derived from emergence.

EMERGENCE SESSION PLANNING

- [] 4. At least one structured emergence session has been scheduled before any design work begins.
- [] 5. The emergence session will use real customer voices: live calls, transcripts, recorded sessions, or direct customer interviews. (Not the client's description of customer needs.)
- [] 6. The designer has confirmed that transcripts or recordings are available, or that a live session can be observed.
- [] 7. The emergence session will not include classification questions in the first exchanges. The transcript will be reviewed for premature collapse before use.

FIVE-METHOD LIFE CYCLE ALIGNMENT

- [] 8. The client has been briefed on the five-method life cycle and understands the sequence: Emergence → Simulation → Discovery → To-Be Selling → Stewardship.
- [] 9. The client has been told, explicitly, that as-is discovery may reveal process change requirements – and that process change may be part of the engagement deliverable.
- [] 10. The client understands that the AI agent will not be deployed until all five methods have been completed to the designer's standard.

PROCESS CHANGE READINESS

- [] 11. The client has a decision-maker in the room who can authorize process changes, not just a technical team that can implement AI features.
- [] 12. The client has confirmed there is organizational appetite for process change if the gap table reveals it is necessary.
- [] 13. The designer has confirmed that deploying an AI agent on the current process – if the current process is broken – is not the engagement deliverable.

SCOPE CONFIRMATION

- [] 14. The engagement scope is defined at the workflow level, not at the feature level. ("Handle cancellations and rescheduling" is a workflow scope. "Build a chatbot with six intents" is a feature scope. These are different engagements.)
- [] 15. The designer has identified which workflow the engagement covers

and confirmed there is one solution statement per workflow.
(Multiple unrelated workflows require separate engagements.)

Pre-Engagement Checklist Complete: Yes No – Item(s) outstanding: _____
Designer Signature: _____ Date: _____

Checklist 2: Method 1 — Emergence Complete

**To be completed after the emergence session(s) and before beginning the blind ideal simulation.
Purpose: confirm that emergence is genuinely complete and the solution statement is ready to drive Method 2.**

METHOD 1 – EMERGENCE COMPLETE CHECKLIST

Engagement: _____ Date: _____
Designer: _____

TRANSCRIPT AND SESSION

- 1. At least one transcript from a real customer-facing session has been collected and reviewed. (Minimum 15-20 exchanges for a meaningful emergence analysis.)
- 2. The transcript has been reviewed for premature collapse moments. All such moments have been noted in the Emergence Worksheet.
- 3. The designer did not accept the first apparent problem statement as the final one. The transcript was read through to the end before a problem statement was drafted.

EMERGENCE MOMENT

- 4. The emergence moment has been identified and annotated on the Emergence Worksheet. A specific line or exchange has been cited.
- 5. Emotional intensity markers have been noted. (Repetition under stress, self-correction, return to the same concern more than once.)
- 6. The designer can articulate what shifted at the emergence moment – what moved from surface complaint to underlying condition.

PROBLEM STATEMENT

- 7. A full problem statement has been written in natural language – in the speaker's frame, not the designer's frame.
- 8. The problem statement has not been pre-cleaned into workflow

terminology. It sounds like a person, not a process document.

- [] 9. The problem statement captures the complete expressed need – including the emotional and contextual dimensions, not just the transactional request.

SOLUTION STATEMENT

- [] 10. A solution statement has been written: one sentence naming the equivalent resolved condition.
- [] 11. The solution statement begins with the structure: "[Actor] experiences [condition] when [trigger]" – or an equivalent formulation that clearly names what success looks like.
- [] 12. The solution statement does not name a process, a system, or a technology. It names an outcome.

GATING WORKSHEET

- [] 13. The Problem Statement = Solution Statement Worksheet (Template 2) has been completed.
- [] 14. Equivalence has been confirmed: if the solution statement were realized, the problem statement would cease to exist.
- [] 15. Template 2 has been signed by the designer.

Method 1 Complete: [] Yes [] No – Item(s) outstanding: _____

Designer Signature: _____ Date: _____

Checklist 3: Method 2 — Blind Ideal Simulation Complete

To be completed after the blind ideal simulation and before beginning as-is discovery. Purpose: confirm that the simulation was run cleanly, the to-be model has been extracted, and the authority chain has been identified.

METHOD 2 – BLIND IDEAL SIMULATION COMPLETE CHECKLIST

Engagement: _____ Date: _____

Designer: _____

SIMULATION INTEGRITY

- [] 1. The simulation was run in a sandbox, demo, or non-production environment. The prompt was not submitted to a system that has access to client data.
- [] 2. The simulation prompt was reviewed before submission and confirmed

free of: client organization name, current process details, system names, staffing or org structure, policy specifics.

- [] 3. The simulation prompt pre-submission checklist (Template 3) was completed before each run.

MULTIPLE RUNS

- [] 4. The simulation was run a minimum of two times with the same prompt.
- [] 5. Both outputs have been saved with run dates and version notes.
- [] 6. Differences between runs have been noted. If the runs produced materially different to-be models, additional runs were conducted and the more robust model was selected with documented rationale.

TO-BE MODEL EXTRACTION

- [] 7. The Transcript Extraction Worksheet (Template 4) has been completed for at least one simulation run.
- [] 8. Every step in the simulation output has been classified as Pattern or Authority. No step has been left unclassified.
- [] 9. The extracted to-be model is coherent: it moves from first customer contact to confirmed resolution, with no logical gaps.

AUTHORITY CHECKPOINTS

- [] 10. Every authority step has been identified and named.
- [] 11. For each authority step, the following have been documented: authority actor, authorization signal, trigger for request, timeout/escalation path.
- [] 12. No authority step has been converted to a pattern step without documented rationale. (Some steps that appear to be pattern steps carry authority requirements that are not obvious.)

STATUS LANGUAGE AND FORBIDDEN CLAIMS

- [] 13. A status label has been assigned to every step in the to-be model.
- [] 14. A draft status language library (Template 10) has been started.
- [] 15. A draft forbidden claims list has been started. Every authority step has generated at least one forbidden claim entry.

TO-BE MODEL DOCUMENT

- [] 16. The To-Be Model Template (Template 5) has been completed.
- [] 17. The solution statement in the To-Be Model Template matches the confirmed solution statement from Template 2 exactly.
- [] 18. Template 5 has been signed by the designer.

Method 2 Complete: Yes No – Item(s) outstanding: _____

Designer Signature: _____ Date: _____

Checklist 4: Method 3 — As-Is Discovery Complete

To be completed after as-is discovery sessions and before beginning the transformation selling conversation. Purpose: confirm that reality has been fully documented — official process, real process, workarounds, authority structure, and failure modes.

METHOD 3 – AS-IS DISCOVERY COMPLETE CHECKLIST

Engagement: _____ Date: _____

Designer: _____

OFFICIAL PROCESS

- 1. The official process has been documented: policy documents, procedure guides, training materials have been reviewed or described by the client.
- 2. The official process has been mapped step by step, with the systems, roles, and timelines named.
- 3. Dates on official documents have been noted. (Documents more than 12 months old without revision may not reflect current practice.)

REAL PROCESS

- 4. The discovery session included direct questions about what actually happens – not just what should happen.
- 5. The designer asked specifically where official process and real process diverge, and received a specific answer.
- 6. The designer interviewed at least one frontline practitioner – someone who executes the process, not just someone who manages or oversees it.

WORKAROUNDS

- 7. At least one informal workaround has been identified and documented. (If none were identified, the discovery session may not have gone deep enough. Revisit Category 3 questions.)
- 8. Each workaround has been traced to the gap it fills: what is broken or missing in the official process that the workaround compensates for?

- [] 9. The tacit knowledge dependencies have been identified: what knowledge lives only in the heads of specific individuals and is not documented anywhere?

AUTHORITY POINTS

- [] 10. Every authority point in the as-is process has been identified: where does something require human approval before it can move?
- [] 11. For each authority point: the named authority actor has been confirmed, and the designer has verified that person is actually available to perform that function.
- [] 12. Authority gaps have been identified: places where no one is clearly the authority, where authority is performed by someone without formal authorization, or where authority steps are routinely skipped.

FAILURE MODES

- [] 13. At least two specific failure modes have been documented – not theoretical failures, but actual failures that have occurred.
- [] 14. At least one customer complaint related to this workflow has been reviewed or described.
- [] 15. The detection mechanism has been assessed: does the organization know when this workflow fails, or does it learn from complaints after the fact?

GAP TABLE

- [] 16. The As-Is vs. To-Be Gap Table (Template 7) has been completed.
- [] 17. Every to-be step has a corresponding entry in the gap table. No to-be step has been left without an as-is equivalent or a "Missing" classification.
- [] 18. Transformation priorities have been assigned to each gap.
- [] 19. The primary transformation requirement – the single gap that most directly blocks the solution statement – has been identified and marked.

CLIENT SIGN-OFF

- [] 20. The as-is map has been reviewed with the client and confirmed as an accurate representation of current reality.
- [] 21. The client has signed off on the as-is reality document. The as-is map is what it is – not an aspirational version of current practice, and not a blame document. It is reality.

Method 3 Complete: Yes No – Item(s) outstanding: _____

Designer Signature: _____ Date: _____

Checklist 5: Method 4 — To-Be Selling Complete

To be completed after the transformation selling conversation and before moving to production design. Purpose: confirm that the client has understood and committed to the transformation requirements — not just approved an AI feature.

METHOD 4 – TO-BE SELLING COMPLETE CHECKLIST

Engagement: _____ Date: _____

Designer: _____

GAP PRESENTATION

- 1. The gap table was presented to the client as a diagnostic, not a criticism. The framing was: "Here is the distance between where you are and where the solution requires you to be."
- 2. The presentation did not imply that the gap is the client's fault. The gap is the difference between as-is reality and to-be ideal. Gap does not mean failure.
- 3. High-priority gaps were clearly distinguished from medium and low priority gaps. The client understands which gaps block the solution statement.

PROCESS CHANGE REQUIREMENTS

- 4. Every gap that requires a process change has been named explicitly. The client has not been allowed to believe that an AI agent can bridge a process gap without process change.
- 5. The designer has stated plainly: "A commodity AI provider automates the current process. This engagement sells movement from as-is toward to-be." The client understands the difference.
- 6. Process changes that require staffing, authority structure, or policy changes have been identified as organizational decisions, not technical decisions.

CLIENT COMMITMENTS

- 7. The client has committed to a specific transformation target: which gaps will be addressed, in which sequence, by which date.
- 8. The transformation target is documented in writing and signed

by a client decision-maker.

- [] 9. The client has confirmed who owns the authority steps in the to-be model. Every authority step has a named human actor who has accepted responsibility for that step.

AUTHORITY AND STATUS LANGUAGE

- [] 10. The authority paths have been agreed: the client has confirmed the authority structure and identified the named actors for each authority checkpoint.
- [] 11. The status language library has been reviewed with the client and agreed. The client understands that the AI agent will use only these status labels – and not improvise language that implies completion.
- [] 12. The forbidden claims list has been reviewed with the client and agreed. The client has confirmed that the listed claims are genuinely forbidden – even when they would be convenient to make.

TRANSFORMATION DOCUMENTATION

- [] 13. The transformation target has been documented in a client-facing summary document that can serve as the basis for a future stewardship review.
- [] 14. A timeline and accountability structure for the transformation has been agreed – who is responsible for each process change, and when it will be complete.
- [] 15. The engagement has confirmed: the AI agent will not be deployed against high-priority missing gaps. It will be deployed against the process that exists after the transformation is complete – or scoped to the steps that can be served now.

Method 4 Complete: [] Yes [] No – Item(s) outstanding: _____

Designer Signature: _____ Date: _____

Checklist 6: Pre-Launch Production Safety

To be completed before any AI agent is deployed to production. Purpose: confirm that the production safety architecture is in place — authority map, forbidden claims, status language, human actor assignments, false completion testing, and monitoring.

PRE-LAUNCH PRODUCTION SAFETY CHECKLIST

Engagement: _____ Date: _____

Designer: _____

AUTHORITY MAP

- 1. The Authority Map (Template 8) is complete. Every authority step in the to-be model has a corresponding row in the map.
- 2. Every authority step has a named, available human actor. Not a role title – a specific person or position that is currently staffed.
- 3. Every authority step has a documented timeout and escalation path. If the authority actor is unavailable, the workflow has a defined next step.

FORBIDDEN CLAIMS AND STATUS LANGUAGE

- 4. The Forbidden Claims Checklist (Template 9) is complete and has been integrated into the AI agent prompt.
- 5. The Status Language Library (Template 10) is complete and has been integrated into the AI agent prompt.
- 6. The AI agent prompt has been reviewed by the designer to confirm that no prompt language contradicts or weakens the forbidden claims or status library constraints.

FALSE COMPLETION TESTING

- 7. A false completion test has been conducted: the AI agent was presented with scenarios designed to elicit forbidden claims, and its responses were reviewed.
- 8. Every forbidden claim on the checklist was tested. The agent did not produce any forbidden language in testing.
- 9. Edge cases were tested: scenarios where the customer applies pressure ("just tell me it's approved"), scenarios where the resolution is ambiguous, and scenarios where the authority actor has not yet acted.

HUMAN ACTOR READINESS

- 10. Every person named as an authority actor has been briefed on the workflow and their role in it. They know what the AI will and will not do, and what they are expected to do when the AI routes to them.
- 11. The handoff from AI to human actor has been tested end-to-end. The human actor receives the right information at handoff.

- 12. There is no authority step that has been configured to be performed by the AI agent. Authority steps are human steps.

INCIDENT MONITORING

- 13. Incident monitoring is in place: a mechanism exists to detect when the AI agent produces a false completion, tool failure, or other anomaly in production.
- 14. The Incident Stewardship Log (Template 11) has been initialized for this workflow. The steward is named and briefed.
- 15. An alert threshold has been defined: how many incidents of what type will trigger an immediate stewardship review?

STEWARDSHIP PLAN

- 16. A stewardship plan exists: who is responsible, at what interval will the workflow be reviewed, what events trigger an out-of-cycle review (model update, policy change, business change, customer behavior shift).
- 17. The steward has acknowledged responsibility in writing.
- 18. The next scheduled stewardship review date has been calendared.

Pre-Launch Checklist Complete: Yes No – Item(s) outstanding: _____

Designer Signature: _____ Date: _____

Client Authorization Signature: _____ Date: _____

Checklist 7: Stewardship Review

To be completed at each scheduled stewardship review and whenever a triggering event occurs.

Purpose: confirm that the workflow has been evaluated against the current reality and a reasoned renewal decision has been made.

STEWARDSHIP REVIEW CHECKLIST

Engagement: _____ Date: _____

Steward: _____ Review Type: Scheduled Triggered

Trigger (if triggered review): _____

RENEWAL AUDIT

- 1. The Renewal Audit Template (Template 12) has been completed for this review cycle. Every section is filled in.
- 2. All changes since the last review have been documented in

Section 2 of the audit: model, business, tools, customer behavior, and policy/regulatory changes.

THE KEY RENEWAL QUESTION

- 3. The key renewal question was asked and answered directly:
"If we designed this workflow today from scratch, would we design it the same way?"
- 4. The answer was not assumed to be yes. It was examined against each documented change.
- 5. If the answer is no or uncertain, the rationale for the resulting recommendation (Renew, Adjust) has been documented.

SOLUTION STATEMENT REVIEW

- 6. The solution statement has been re-read and confirmed as still valid. It has not been silently assumed to be valid because no one raised it.
- 7. If the solution statement is no longer valid, the review has been escalated: this is not an adjustment. This is a new workflow design beginning at Method 1.

SIMULATION RERUN

- 8. If the AI model has been updated since the last simulation, a simulation rerun has been conducted or has been explicitly decided against with documented rationale.
- 9. If a rerun was conducted, the output has been compared to the current to-be model and material differences have been noted.
- 10. If the simulation produced a materially different to-be model, the difference has been assessed for impact on the solution statement.

TRANSFORMATION TARGET REVIEW

- 11. The transformation target from Method 4 has been reviewed:
have the committed process changes been implemented?
- 12. If process changes remain outstanding, their impact on current deployment has been assessed and documented.
- 13. If the business has changed in ways that alter the transformation target (new product lines, new customer segments, changed scope), this has been noted as a potential renewal trigger.

INCIDENT HISTORY REVIEW

- 14. All incidents since the last review have been reviewed. Patterns

have been identified. (Multiple incidents of the same type indicate a systemic issue, not isolated failures.)

- [] 15. Any open incident entries in the Stewardship Log have been escalated and are being addressed before this review closes.

RECOMMENDATION AND NEXT STEPS

- [] 16. A recommendation (Renew, Adjust, or Keep) has been made and documented with justification.
- [] 17. If Renew: a new engagement has been scoped, beginning at Method 1.
- [] 18. If Adjust: specific adjustments have been identified, documented, and assigned to an owner with a completion date.
- [] 19. If Keep: the decision and its rationale have been documented. "Keep" is not the default – it is a finding.
- [] 20. The next review date has been set and calendared.

Stewardship Review Complete: [] Yes [] No – Item(s) outstanding: _____

Steward Signature: _____ Date: _____

Key Takeaways

- Checklists gate transitions, not tasks. Each checklist above is associated with a phase exit — a moment where the designer formally confirms that the work of this phase is done and the next phase may begin. Using them only at the end misses their purpose. Use them as you work.
- The pre-engagement checklist is often the most skipped and the most consequential. Engaging with a client who has not been told that process change may be required is an engagement with a structural problem built in. Find out early. Early patience prevents late repair.
- The Pre-Launch Production Safety Checklist is the designer’s last line of defense before consequences become real. Item 12 — “There is no authority step that has been configured to be performed by the AI agent” — is the item most often under pressure. Hold it.
- The Stewardship Review Checklist is the professional signature on a living design. Signing it means: I have looked, I have asked the hard question, and I have recommended based on evidence. A workflow that has not been through this process is not stewarded. It is merely running.
- The goal is not fewer checklists. The goal is fewer wrong workflows. These checklists are the instrument of that goal. # Conclusion

What the Designer Carries Forward

AI workflow design begins with a human speaking. Everything else follows from that.

Not from the technology. Not from the client brief. Not from the process diagram on the conference room wall. A human speaks — imprecisely, emotionally, unfinished — and the discipline of AI workflow design begins in the act of listening. The designer who can hold that moment, who refuses to collapse it into a category before it has fully arrived, is the designer who will build something worth building. The designer who routes too early builds something efficient and wrong.

This is the central thesis of everything in this book, and it is worth saying plainly at the end: the entire five-method life cycle exists to protect the emergence of what is real from the pressure of what is convenient. Emergence gives you the solution. Everything that follows is the work of building toward it, measuring the distance, and closing the gap.

The five methods form a spine. They are not a checklist to complete and set aside — they are the posture of the designer across the full life of the workflow.

Emergence holds the space for a human being to arrive fully. It resists the pull of premature classification. It produces the problem statement and, from the problem statement, the solution statement. The solution statement is the one fixed point in the entire design process. Everything else can change. The workflow can be redesigned. The tools can be replaced. The process steps can be reordered. But the solution statement is sacred. If you abandon it, you are no longer designing toward the same destination.

The blind ideal simulation asks: what would this workflow look like if reality imposed no constraints? Hallucination gives you the ideal. In production, hallucination is false completion — a failure, a risk, a thing to be designed against. In discovery, hallucination reveals the ideal. The designer who understands this distinction uses the AI's generative nature as a precision instrument rather than treating it as a liability to be suppressed.

As-is discovery gives you reality. The as-is model is not destiny; it is the starting point. But it must be faced honestly, because a workflow designed on a false picture of reality will fail the moment it meets the real process. Do not contaminate the to-be model too early. And do not let the as-is model become an excuse for low ambition. The gap table is not a list of complaints. It is a map of the distance between where the organization is and where the solution requires it to be.

To-be selling gives you transformation. A commodity AI provider automates the current process. A serious AI workflow designer sells movement from as-is toward to-be. This is the distinction that defines professional practice. It requires the designer to be willing to tell a client that the technology is not the problem — the process is the problem, and the technology will only make the process faster, not better, unless the process changes too.

Stewardship gives you renewal. The workflow does not end at launch. It begins at launch. Maintenance preserves the workflow. Stewardship renews the workflow around the solution. The steward who asks — at every review, without exception — “if we designed this today from scratch, would we design it the same way?” is the steward who keeps the workflow alive. The steward who never asks that question is tending a relic.

What the designer carries forward is a disposition, not a procedure.

The disposition is this: be willing to obsolete your own design. When the solution statement points to a better path, follow it, even if following it requires dismantling something you built with care. The workflow is not sacred. The solution statement is sacred. A designer who has internalized this will never defend a bad workflow simply because they authored it. The second-best designer is the one who understands why this matters — who can articulate the principle even if they find it difficult to act on. The best designer acts on it without hesitation.

The designer also carries forward a relationship to AI that this book has tried to establish from the first chapter: AI as translator, living bridge, pattern-bearing participant. Not a tool that executes instructions. Not a chatbot that fields questions. Not a search engine with a voice. Something more structurally interesting than any of those framings — an entity that can hold the shape of an ideal without the weight of the actual, that can traverse the distance between problem statement and solution statement faster than any prior technology, and that fails in ways that are, if the designer is paying attention, instructive.

The designer who leans into the nature of AI — not around it, not despite it, into it — is the designer who builds workflows that are worthy of the discipline described in these pages. Lean into emergence, because AI can hold emergence without impatience. Lean into the ideal simulation, because AI can generate ideals without attachment. Lean into status language and authority design, because AI, precisely because it is capable of false completion, reveals exactly where the human authority structure must be intact.

The speaking human at the beginning of the workflow is unfinished. Every speaking human arrives unfinished. The designer's job is not to finish them quickly. The designer's job is to build a workflow that receives them fully, moves them accurately through a process that has been designed against the ideal and measured against reality, delivers them to a resolved condition that was named at the beginning, and renews itself when the world changes.

That is AI workflow design. That is what you carry forward. # Glossary

Terms as Used in This Textbook

AI workflow design The discipline of designing, deploying, and stewarding AI-assisted workflows from first customer contact to confirmed resolution. AI workflow design is distinct from software development (which builds systems) and prompt engineering (which configures models) in that it begins with human emergence, derives a solution statement, and builds all workflow architecture in service of that statement.

AI-native workflow A workflow designed from the ground up to incorporate AI as a structural participant — not a workflow that has had AI features added to it. An AI-native workflow uses emergence, pattern-authority classification, status language, and authority mapping as foundational design elements, rather than retrofitting these elements onto a pre-existing process.

Emergence The process by which a human speaker's full problem — including its emotional, contex-

tual, and relational dimensions — arrives through sustained, unhurried conversation. Emergence is not extraction. Extraction assumes the problem is already formed and simply needs to be retrieved. Emergence assumes the problem is forming as the speaker speaks, and that the speaker’s full need cannot be known until it has had room to arrive.

Emergence session A structured session — live conversation, recorded call review, or transcript analysis — in which the designer holds the space for emergence to occur without premature classification, routing, or collapse. An emergence session produces a problem statement and, from it, a solution statement. It is the first act of the five-method life cycle.

Premature collapse The act of routing a customer’s expressed need into a predefined category before the full problem has been allowed to emerge. Premature collapse produces workflows that are efficient and wrong: they route the customer correctly according to the surface expression of their need while failing to address the underlying condition. The cost of premature collapse appears not at intake but at resolution.

Problem statement The full expression of what the speaker needs, in natural language, as it arrives through emergence. A problem statement is not a ticket category, a classification label, or a summary written in workflow terminology. It is a paragraph — sometimes an uncomfortable, emotionally loaded paragraph — that captures the speaker’s complete expressed need. The problem statement is the raw material from which the solution statement is derived.

Solution statement A single sentence naming the equivalent resolved condition: the state of affairs that would exist if the problem statement were fully resolved. The solution statement is the fixed point of the entire design process. Workflows may be redesigned; tools may change; processes may evolve. The solution statement does not change unless the underlying human need has changed. It is the one thing that is sacred.

Pattern (as distinct from authority) A step in a workflow that can be completed by an AI agent or automated system without requiring human authorization, legal sign-off, or policy decision. Pattern steps are deterministic given correct inputs: confirm receipt, route a request, generate a draft, look up a status. They are the steps AI executes reliably and without authority risk. Distinguishing pattern steps from authority steps is the foundational act of workflow architecture.

Authority A step in a workflow that requires a human actor to exercise judgment, make a binding decision, or authorize an outcome. Authority steps cannot be delegated to AI without creating an authority gap. Examples include: approving a claim, determining legal eligibility, authorizing a refund, issuing a binding agreement. Every authority step must have a named human actor, an authorization signal, and a timeout/escalation path.

False completion A statement made by an AI agent that claims or implies a completed outcome that has not yet been authorized by the appropriate authority actor. False completions are the most consequential failure mode in AI workflow design. They are not lies — the AI is not being deceptive. They are structural failures: the workflow was not designed with authority steps properly separated from pattern steps, and the AI, completing a pattern, claimed an authority outcome. “Your refund has been processed”

is a false completion when a billing authority has not yet reviewed the case.

Status language A controlled vocabulary of status labels used by an AI agent to report the accurate state of a transaction at each point in the workflow. Status language is the linguistic equivalent of the authority map: it ensures that what the AI says corresponds precisely to what has actually been authorized. Status labels include: Drafted, Prepared, Submitted, Routed, Pending Verification, Verified, Approved, Issued, Confirmed, Completed. Each label has a precise meaning, a trigger event, and a set of permitted and forbidden AI language.

Authority map A structured document that records every authority checkpoint in a workflow: the step number, step description, the named authority actor, the authorization signal, the trigger for the authorization request, the timeout and escalation path, the forbidden claim at that step, and the status label. The authority map is a production safety document. It is integrated into the AI agent prompt and reviewed at every stewardship cycle.

Forbidden claims A list of specific statements the AI agent must never make in a given workflow, because those statements assert outcomes that have not been authorized by the relevant authority actor. Forbidden claims are workflow-specific: what is forbidden in a claims workflow differs from what is forbidden in an appointment scheduling workflow. The forbidden claims list is built from the authority map and updated whenever a new false completion incident is detected in production.

Blind ideal workflow simulation (hallucination in discovery mode) The use of an AI language model's generative capacity — unconstrained by as-is reality — to produce an ideal-state workflow against a confirmed solution statement. In this mode, the AI's tendency to generate plausible, complete scenarios is a design asset rather than a risk. The simulation is run in a sandbox, with no client-specific process data in the prompt, so that the AI generates the ideal without contamination from the actual. In production, hallucination is false completion. In discovery, hallucination reveals the ideal.

To-be model The ideal-state workflow derived from the blind ideal simulation and structured through the extraction process. The to-be model describes how the workflow should function when fully realized — step by step, with pattern and authority classification, status labels, authority actors, and forbidden claims. The to-be model is the benchmark against which as-is reality is measured.

As-is model The documented reality of how the workflow currently functions, derived through structured as-is discovery. The as-is model includes the official process, the real process, the informal workarounds, the authority structure (including authority gaps), and the known failure modes. The as-is model is reality, not destiny. It is the starting point of transformation, not the ceiling of ambition.

As-is discovery The structured investigation of the current process through direct interview, observation, and document review, organized across five discovery categories: official process, real process, informal workarounds, authority points, and failure modes. As-is discovery is Method 3 in the five-method life cycle. It must be conducted after the to-be model is complete, so that as-is reality does not contaminate the ideal.

Gap table The analytical document that maps every step in the to-be model against its as-is equivalent and classifies the gap between them. Gap types are: None (as-is matches to-be), Partial (as-is step

exists but is incomplete or inconsistent), Workaround (as-is step is performed via informal or unsupported means), or Missing (no as-is equivalent exists). The gap table is the primary instrument of the transformation selling conversation.

Transformation target The specific set of changes — process, policy, authority structure, system — that the client commits to implementing in order to move from as-is reality toward the to-be ideal. The transformation target is the output of Method 4. It is documented, signed, and used as the baseline for stewardship review. The AI agent is deployed against the process that exists after the transformation target is partially or fully implemented, not against the unchanged as-is process.

Commodity AI provider An AI vendor or implementation approach that automates the current process without interrogating whether the current process serves the solution statement. A commodity AI provider takes the as-is process as given and makes it faster. A serious AI workflow designer takes the solution statement as the fixed point and builds toward it — which may require challenging, changing, or replacing the as-is process entirely.

Stewardship The ongoing practice of maintaining a workflow’s alignment with its solution statement across the full life of the deployment. Stewardship is not maintenance. It involves active, regular examination of whether the workflow still serves the solution statement — and willingness to renew the design when it does not. Stewardship gives you renewal.

Maintenance (as distinct from stewardship) The routine operational work of keeping a deployed workflow running: updating integrations, fixing bugs, adjusting prompts for minor model drift, responding to individual incidents. Maintenance preserves the workflow. Stewardship renews the workflow around the solution. Both are necessary; neither substitutes for the other.

Incident stewardship The practice of recording, classifying, investigating, correcting, and learning from individual incidents of workflow failure in production. Incident stewardship produces the Incident Stewardship Log, which is the institutional memory of the workflow’s failure history. Patterns in the incident log are the primary signal for a triggered stewardship review.

Renewal stewardship The practice of conducting regular, structured reviews of the entire workflow — its solution statement, its to-be model, its authority structure, and its safety documents — at scheduled intervals and in response to triggering events. Renewal stewardship asks the key question: “If we designed this today from scratch, would we design it the same way?” If the answer is no or uncertain, a renewal is required.

Renewal audit The formal document produced during a renewal stewardship review. The renewal audit covers: confirmation of the solution statement, documentation of all changes since the last review, the key renewal question and its answer, a simulation rerun if indicated, findings, and a recommendation (Renew, Adjust, or Keep). The renewal audit is signed by the steward and filed with the engagement record.

Containment (as distinct from correction) The immediate action taken when a workflow incident is detected in production — designed to prevent further harm before the root cause is investigated. Containment may include taking an agent offline, adding temporary override language to a prompt, or manually

routing affected cases to human handlers. Containment is not correction: it stops the bleeding. Correction addresses the root cause and implements a durable fix.

Sound wave concept The framing of a customer’s expression — across a conversation, a call, or a session — as a sound wave: it begins before the first word, continues through the silences and self-corrections, and does not resolve until the full expression has been heard. The sound wave concept is the theoretical basis for the emergence method. It is why premature collapse is structurally harmful: it cuts the sound wave short before the signal has fully arrived.

Living bridge A framing of the AI agent’s role in a workflow: not as a fixed automation executing a script, but as a living bridge between the human’s expressed need and the system’s capacity to resolve it. The living bridge holds the problem statement without judgment, traverses the pattern steps without false completion, routes to authority steps without bypassing them, and delivers status language that is accurate rather than reassuring.

Pattern-bearing participant A framing of the AI’s nature in workflow design: an entity that carries and recognizes patterns — across language, across human expressions of need, across workflow steps — and participates in the workflow as a structural actor rather than a passive conduit. The pattern-bearing participant does not merely respond to inputs; it holds the shape of the workflow and moves the transaction through it.

The equation (problem statement = solution statement) The foundational relationship in AI workflow design: the problem statement and the solution statement are equivalent. That is, the solution statement — if realized — would fully resolve the problem statement. This equivalence must be confirmed before design work proceeds. If the solution statement does not fully resolve the problem statement, emergence is not complete. The equation is not arithmetic; it is a structural claim about the relationship between expressed need and designed response. # Canonical Lines of AI Workflow Design

A Collected Reference

These are the defining statements of the framework presented in this textbook. They are not slogans. Each line is a precise claim about how AI workflow design works — what it requires, what it guards against, and what it makes possible. They are collected here for use in practice, in teaching, and in the moments during an engagement when a designer needs the framework’s spine in a single sentence.

Emergence gives you the solution.

The solution statement is not derived from analysis, client briefs, or process documentation. It is derived from emergence — from the act of listening to a human speak long enough for their full need to arrive. The designer who skips emergence, or collapses it too early, does not have a solution statement. They have an assumption dressed as a solution.

Hallucination gives you the ideal.

This line only makes sense within the specific context of the blind ideal simulation. In that context — with the AI running in demo mode, unconstrained by as-is reality — the AI’s tendency to generate plausible, complete, ideal-state scenarios is a design instrument. The designer harvests it. Outside that context, in production, hallucination is a failure mode to be designed against.

As-is discovery gives you reality.

The as-is model is not built from what the organization says it does. It is built from what the organization actually does — including the informal workarounds, the authority gaps, the tacit knowledge that lives only in specific people’s heads, and the failure modes the organization has learned to live with. As-is discovery is the act of facing that reality without flinching.

To-be selling gives you transformation.

The gap between as-is and to-be is not a complaint. It is a map. The transformation selling conversation — Method 4 — is the act of presenting that map to the client and asking them to commit to closing it. Not to automate the current process. To move toward the to-be ideal. This is what distinguishes a professional AI workflow designer from a commodity AI vendor.

Stewardship gives you renewal.

A workflow deployed and never reviewed is a workflow aging in silence. Stewardship is the practice that keeps the workflow alive — not by preserving it, but by re-earning its alignment with the solution statement at regular intervals. Stewardship is what happens after launch. It is not maintenance. It is renewal.

The workflow is not sacred. The solution statement is sacred.

Every design decision in a workflow is conditional: it can be reconsidered, revised, or replaced. The solution statement cannot. It is the fixed point from which all design decisions are measured. A designer who defends a workflow design in the face of evidence that a better path to the solution statement exists has confused the means for the end.

Maintenance preserves the workflow. Stewardship renews the workflow around the solution.

These are not synonyms. Maintenance keeps the current workflow functioning. Stewardship asks whether the current workflow should still exist — or whether the solution statement now points to something different. A workflow can be perfectly maintained and completely outdated. Stewardship is the practice that catches the difference.

In production, hallucination is false completion. In discovery, hallucination reveals the ideal.

The same property of AI — the capacity to generate complete, plausible scenarios — is a liability in one context and an instrument in another. The designer who understands this distinction uses AI in the way its nature makes possible, rather than treating its generative capacity as uniformly dangerous. Context determines whether generation is a risk or a resource.

The beginning of the workflow is not extraction. The beginning of the workflow is emergence.

Extraction assumes the problem is fully formed and simply needs to be retrieved. Emergence assumes the problem is forming as the speaker speaks. These produce different first steps, different early AI behaviors, and different workflow designs. A workflow that begins with extraction will classify too early, route too quickly, and miss the full problem. Emergence is not slower. It is more accurate.

Every speaking human arrives unfinished.

No human being who contacts a support line, a clinic, a law firm, or any other organization arrives with a fully formed, precisely articulated statement of their need. They arrive in process. They are working out what they need as they speak. The designer who builds a workflow that can receive an unfinished human — and hold the space for them to finish — has built something the classifier cannot.

Do not collapse the human too early.

Premature collapse is the most common failure in AI workflow first-contact design. It looks efficient. It feels helpful. It is wrong. The cost does not appear at intake — it appears at resolution, when the workflow has been routing a half-formed problem to a solution that does not fit. Hold the emergence longer than feels necessary. The accuracy is worth it.

Do not contaminate the to-be model too early.

The blind ideal simulation must be run before as-is reality enters the design process. Once the designer knows what the organization currently does, that knowledge exerts gravitational pull on the design. The designer begins building toward what is possible, not what is ideal. Run the simulation first. Protect the ideal from the actual until the ideal is documented and fixed.

The as-is model is reality, not destiny.

The as-is discovery process reveals what is true about the current process — including its gaps, its workarounds, and its failures. This is not a finding about what is possible. It is a finding about where transformation must begin. The as-is model sets the starting line. The solution statement sets the destination. The transformation target names the route.

A commodity AI provider automates the current process. A serious AI workflow designer sells movement from as-is toward to-be.

This is the distinction that defines the profession. A vendor who takes the as-is process as given and makes it faster is providing a commodity. A designer who takes the solution statement as fixed and builds toward it — even when that requires the client to change their process — is providing transformation. The client may not know the difference at first. They will know the difference at implementation.

Be willing to obsolete your own design.

When the solution statement points to a better path — a new model, a changed process, a redesigned workflow — follow it. Authorship is not a reason to defend an outdated design. The designer who is willing to rebuild what they built is the designer whose work stays aligned with what the work is for.

If we designed this today from scratch, would we design it the same way?

This is the key renewal question. It is asked at every stewardship review. It is not rhetorical. A “yes” is a finding — the workflow remains current. A “no” is an instruction — a renewal is required. An “I’m not sure” is also an instruction — run the simulation again and look carefully at what emerges.

The goal is not fewer words. The goal is fewer wrong workflows.

AI workflow design is not optimized for brevity, simplicity, or reduced token count. It is optimized for accuracy: does the workflow reliably move the human from their expressed need to the equivalent resolved condition? A long workflow that does this is better than a short workflow that does not. The templates and checklists in this book are detailed because detail is what accuracy requires.

Early patience prevents late repair.

The moments in the design process that feel slow — the emergence session that goes longer than expected, the discovery interview that surfaces a structural problem, the renewal audit that concludes a full redesign is needed — are the moments that prevent the much greater cost of discovering these problems after deployment. The designer who is impatient early will be very busy late.

Lean into the nature of AI.

Not around it. Not despite it. Into it. AI is a pattern-bearing participant, a living bridge, a translator that can hold emergence without impatience and traverse pattern steps without authority. Its properties — including the ones that can fail — are design information. The designer who leans into the nature of AI builds workflows that are worthy of it.